

MySQLAdminPro

WXES 3182

2005/2006

Name: Wong Teng Foong

Matrics No: WEK 030252

Supervisor: Mr. Mohd Hairul Nizam

Moderator: Mrs. Rodina Ahmad

Abstract

All final year undergraduates are required to carry out the final year project (thesis) which is divided into two phases. The first phase includes more on initial research, planning and study on the particular subject of field chosen. The second phase of the thesis is more emphasized on coding and development of system

Each undergraduate will be assigned to a lecturer who acts as a supervisor responsible in advising and guiding the progress of the overall project. The undergraduate gets to choose the supervisor by either accepting the supervisor's thesis question or proposing his own question to the chosen lecturer

My responsibility is to develop a tool which helps to manipulate the MySQL database. This tool helps to solve the unfriendly command line interface of MySQL by presenting user interfaces on a browser where buttons and links are used to operate the database. Other than that, the tool also equipped with a compressed backup function where it can backup databases in a compressed text file script format

Acknowledgment

The whole process of system development will not be completed without helps and guidance from Mr. Hairul Nizam Md Nasir, my supervisor. He really assists me a lot in the whole process by explaining to me on how the overall system should include. Other than that, he also taught me a lot on documentation and spends so much time in helping me to check my progress to keep me on the track. I believe that this project will not be fulfilled without his help and Mr. Hairul Nizam Md Nasir deserves the credits

Lots of gratitudes to Mrs. Rodina Ahmad and Mrs. Siti Hafizah Ab. Hamid for willing to be my two panels during my viva. Although I did not have much chance to interact with them, but all their comments and remarks during my viva really helps me a lot to enhance my project and most importantly, to improve myself

Lastly, I would like to thank all my friends for their opinions and views on my project. A lot of appreciation to those who had participated in my survey for the system analysis chapter as it served as a basic estimation of requirement from the public

Table of Contents

Abstract	i
Acknowledgement	ii
Table of Contents	iii
List of Figures	viii
List of Tables	ix

Chapter 1: Introduction	1
1.1 Project overview	2
1.2 Problem statement	2
1.3 Objectives	2
1.4 Scope	4
1.5 Research methodology	5
1.5.1 Literature review	5
1.5.2 Study and observation of current system	5
1.5.3 Survey	5
1.5.4 Interview	6
1.6 Development methodology	7
1.6.1 Waterfall model	7
1.7 Expected outcomes	8
1.8 Project timeline	9
1.9 Summary	10

Chapter 2: Literature Review	11
2.1 Background study	12
2.2 Introduction to database	12
2.3 Database management system (DBMS)	12

2.4	Relational database management system (RDBMS)	
2.4.1	Comparison between RDBMS	16
2.5	Structured Query Language (SQL)	18
2.5.1	SQL standard	18
2.5.2	SQL commands reference	18
2.6	MySQL	24
2.6.1	MySQL features	24
2.6.2	MySQL datatypes	25
2.6.3	MySQL supported functions	29
2.6.4	MySQL optimization	41
2.7	Existing tools to manage MySQL	44
2.7.1	Query Browser	45
2.7.2	MySQL Control Center	46
2.7.3	phpMyAdmin	47
2.7.4	Comparison between tools	49
2.7.5	MySQLAdminPro features improvement	50
2.8	Summary	51

Chapter 3: System Analysis **52**

3.1	Introduction	53
3.2	Analysis of survey	53
3.3	Requirements	57
3.3.1	Functional requirements	57
3.3.1.1	Use case description	59
3.3.2	Non-functional requirements	67
3.3.3	Hardware requirements	69
3.3.4	Software requirements	69

Chapter 4: System Design **70**

4.1	Introduction	71
4.2	Architecture design	71
4.2.1	The presentation layer	72
4.2.2	The business logic layer	72
4.2.3	The data layer	73
4.3	Process design	73
4.3.1	Context diagram	74
4.3.2	Level 0 DFD	75
4.3.3	Level 1 DFD	76
4.3.4	Class diagram	78
4.3.5	Sequence diagram	80
4.4	Input and output design	85
4.4.1	Input design	85
4.4.2	Output design	86
4.5	Interface design	86

Chapter 5: System Implementation **90**

5.1	Overall system development approach	91
5.2	Style and design approach	92
5.3	Implementation with JavaScript	93
5.4	Implementation with IntelliJ IDEA Integrated Development Tool (IDE)	93
5.4.1	Program commenting	94
5.4.2	Exception handling	95
5.4.3	Import libraries	95
5.4.4	Apache Struts framework implementation	96
5.5	Implementing the database	96
5.5.1	Setting up the database	96

5.5.2	Database connection	97
5.5.3	Querying the database	98
5.6	System modules and functionalities	98
5.6.1	Access authentication	98
5.6.2	Database and table manipulation	99
5.6.3	User's history logging	100
5.7	Summary	101

Chapter 6: System Testing 102

6.1	Objectives of testing	103
6.2	Test strategies	103
6.2.1	Unit testing	104
6.2.2	Data validation testing	105
6.2.3	Integration testing	105
6.2.4	System testing	105
6.2.5	User acceptance testing	107
6.3	Test plan	103
6.3.1	Unit testing	108
6.3.2	Data validation testing	111
6.3.3	Integration testing	111
6.4	Summary	113

Chapter 7: System Evaluation and Conclusion 114

7.1	Introduction	115
7.2	System strengths	115
7.2.1	Security and privileges	115
7.2.2	Conveniences in database, table and record manipulation	116

7.2.3	Complete database backup and compression	116
7.2.4	Database table export	116
7.2.5	User history logging	117
7.3	System limitations	117
7.3.1	SQL query execution	117
7.3.2	Limited choice of file type to export	117
7.4	Future enhancements	118
7.4.1	Multiple SQL queries execution	118
7.4.2	Multiple choice of file type to export	118
7.5	Problems encountered and solution	118
7.5.1	Settings and configuration difficulties	118
7.5.2	Improperly defined project scope	119
7.5.3	Lack of language mastery	119
7.6	Knowledge and experience gained	120
7.6.1	Improvement in system development practices	120
7.6.2	Improvement in programming skill	120
7.6.3	Project planning	121
7.7	Reviews on goals	121
7.8	Conclusion	122
References		123
Appendix A		124
User manual		126

List of Figures

Figure 1.1	Waterfall Model	7
Figure 1.1	Estimated Work Schedule	9
Figure 1.3	Gantt chart for project timeline	10
Figure 2.1	MySQL command line interface	44
Figure 2.2	Query Browser	46
Figure 2.3	MySQL control center	47
Figure 2.4	phpMyAdmin	48
Figure 3.1	Difficulties in using MySQL command line interface	54
Figure 3.2	Tools which have been used before	55
Figure 3.3	Weaknesses of the tool chosen	56
Figure 3.4	Use case diagram	58
Figure 4.1	Context diagram	74
Figure 4.2	Level 0 DFD	75
Figure 4.3	Level 1 DFD (Authenticate user)	76
Figure 4.4	Level 1 DFD (Update system / user information)	77
Figure 4.5	Level 1 DFD (Conduct user's operation)	77
Figure 4.6	Class diagram	78
Figure 4.7	User Log In Sequence	80
Figure 4.8	Create New Database Sequence	81
Figure 4.9	Insert New Record Sequence	82
Figure 4.10	Drop Database Sequence	84
Figure 4.12	Log in menu	88
Figure 4.13	Main menu	89
Figure 5.1	Program Commenting	94
Figure 5.2	Import Libraries	95
Figure 5.3	Database Connection	97

List of Tables

Table 2.1	Comparison between RDBMS	17
Table 2.2	SQL commands reference	19
Table 2.3	MySQL datatypes	25
Table 2.4	MySQL supported functions	29
Table 2.5	Comparison between tools	49
Table 2.6	MySQLadminPro features improvement	50
Table 3.1	Importance of the features of MySQL tool	56
Table 3.2	Hardware requirements	69
Table 3.3	Software requirements	69
Table 5.1	User Privileges	98

Chapter 1:

Introduction

Chapter 1: Introduction

1.1 Project overview

MySQL Admin Pro is a web application tool to handle and manipulate the MySQL relational database management system. The main idea is to use simple interfaces on the web browser to control the database with Structured Query Language (SQL).

1.2 Problem statement

MySQL is a popular open source relational database management system which operates widely either for personal or business use. MySQL database is well-known of its reliability and performance but it has a huge drawback where queries and operations on the database have to be done in command line mode. This is a difficulty for those who are not familiar with the Structured Query Language (SQL) command and it's quite confusing to view the database in the command prompt mode. Other than this, presently MySQL lacks the feature to backup the entire database in a compressed file.

1.3 Objectives

The objective of this project is to build a web application called the MySQLAdminPro which can manage the MySQL database using the web browser as the client. This is because the original MySQL database comes with the command line interface and by

using this tool, it is able to control and manipulate the database more easily and hassle-free. The resulting system will help users with:

- **Buttons and links interface**

User interface is displayed on the browser along with some common SQL command buttons and this is very useful for those who are not familiar with the SQL syntax

- **SQL query/queries command line interface**

Other than easy to use interface, the MySQLAdminPro provides the user with the command line interface as well so that user may have the flexibility to issue more complicated query/queries.

- **Data manipulation efficiency**

The process of manipulating the database is shown clearly to the user. User can see what they are doing and view the database in an organized way rather than outputting it in the command prompt. This will speed up the time to operate the database and increase efficiency

- **Access authentication**

User may need to enter their username and password to use the MySQLAdminPro tool and this is for security purposes. Unauthorized user may not use the tool unless they register themselves first before they can view the database.

- **Database backup and compression**

User may back up their database in a zip file format. This will help for back up purposes and to compress the size of the database.

1.4 Scope

The MySQLAdminPro is a web-based application where it can only be used to manipulate the MySQL database. It provides features using the SQL keywords and the features that MySQLAdminPro provides are as below:

- **Data retrieval**

User may retrieve data from the database using convenience buttons which provides operations with the SELECT keyword. The SELECT keyword can be used with other common keywords like FROM, WHERE, GROUP BY, HAVING and ORDER.

- **Data manipulation**

User may manipulate the database with buttons which provides operations with the INSERT, UPDATE, MERGE, DELETE and TRUNCATE keywords. These keywords will insert values, update values, combine values, remove row records and delete all data from a table respectively

- **Data definition**

User may create and delete objects like databases and tables, the CREATE and DROP buttons will be provided

Other than the SQL syntax, MySQLAdminPro may provide other features like:

- Compressing the database in a zipped file format
- Access authentication
- Export database to other file format (text or Excel)

- View MySQL information like runtime information and user's past operations history

1.5 Research methodology

1.5.1 Literature review

Subjects concerning the background or detailed information for certain issues will be gathered from articles and documentation which can be downloaded from the internet and as well as books. The materials which are collected will be studied and analyzed to get all the relevant information for the project

1.5.2 Study and observation of current system

Analysis of a few well known systems which is similar to MySQLAdminPro will be conducted and the strengths and weaknesses of those tools will be evaluated. Improvements and development of MySQLAdminPro will be made by adopting the advantages of other tools and improve on its disadvantages

1.5.3 Survey

Survey is a method to gather information from the public randomly. It is one of the best ways to get general understanding of some specific issues so that consideration on what users really need and want will be evaluated. Questions will be set initially and

distributed randomly to public. The responses will then be gathered and analyzed by charts or tables for further assessment.

1.5.3 Waterfall model

1.5.4 Interview

Interview is conducted to gather specific and essential information regarding certain issues. It is usually carried out with someone who is experience or knowledgeable on the subject. Interviews can be conducted either in closed or open manner. In closed interviews, interviewer will look for answers to a pre-defined set of questions while in an open interview, there is no pre-defined agenda and the interviewer will discuss in an open-ended way with the stakeholders.

1.6 Development methodology

1.6.1 Waterfall model

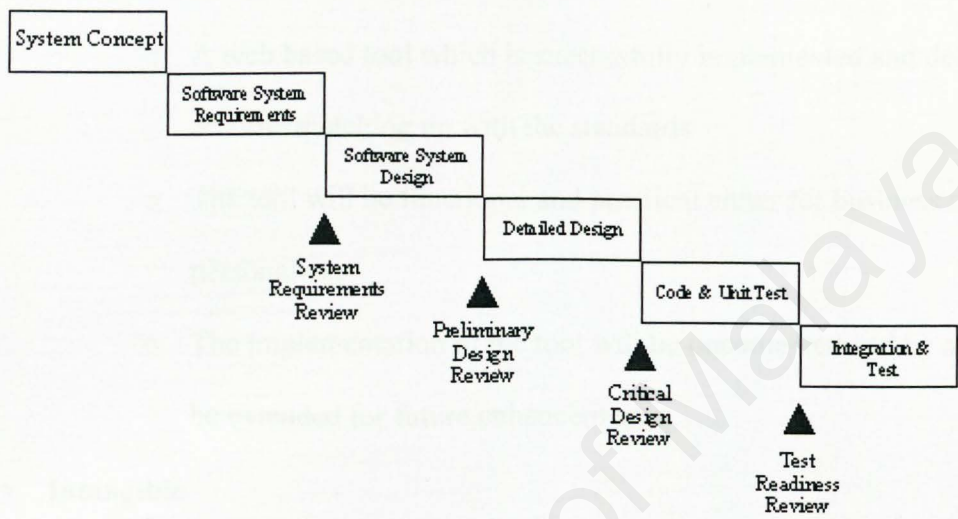


Figure 1.1 Waterfall model

The waterfall model was born in 1970s where it was used for aerospace and defense projects. This model can be described as a step-by-step model where one phase ended and another began once a milestone had been reached (Hoffer and Valacich, 2005). It is suitable for small project where it is easy to manage and the requirements are very well understood. All deliveries and results are happen at the end when the system is ready and documents will be available before that.

1.7 Expected outcomes

The expected outcomes from this project can be viewed in two key areas which are tangible and intangible:

- **Tangible**

- A web based tool which is successfully implemented and delivered on time matching up with the standards
- The tool will be functional and practical either for business or personal use
- The implementation of the tool will be documented and be able to be extended for future enhancement

- **Intangible**

- The use and importance of Relational Database Management System (RDBMS) will be understood easily by new users
- The work load of users who depend heavily on MySQL will be decreased due to the efficiency of the tool
- Creates another alternative for users to choose on MySQL open source tools and realize the benefits of it

1.8 Project timeline

Project timeline is a planned schedule that records the tasks which need to be accomplished and the time allocated to them. It is a good way to keep track of the project development and it is vital to deliver the project on time. Following is the estimated time frame for the tasks in the development of MySQLAdminPro.

















		Task Name	Duration	Start	Finish	Predecessors
1		Project planning	3 days	Mon 8/1/05	Wed 8/3/05	
2		Find appropriate articles	14 days	Thu 8/4/05	Wed 8/17/05	1
3		Chapter 1: Introduction	7 days	Thu 8/18/05	Wed 8/24/05	2
4		Chapter 2: Literature review	21 days	Thu 8/25/05	Wed 9/14/05	3
5		Milestone 1: Literature review	0 days	Wed 9/14/05	Wed 9/14/05	4
6		Preliminary system analysis	14 days	Thu 9/15/05	Wed 9/28/05	5
7		Detailed system analysis	14 days	Thu 9/29/05	Wed 10/12/05	6
8		Milestone 2: System analysis review	0 days	Wed 10/12/05	Wed 10/12/05	7
9		Preliminary system design	14 days	Thu 10/13/05	Wed 10/26/05	8
10		Detailed system design	14 days	Thu 10/27/05	Wed 11/9/05	9
11		Milestone 3: System design review	0 days	Wed 11/9/05	Wed 11/9/05	10
12		Implementation	60 days	Thu 11/10/05	Sun 1/8/06	11
13		System testing	56 days	Wed 11/16/05	Tue 1/10/06	
14		System evaluation	3 days	Wed 1/11/06	Fri 1/13/06	13
15		Milestone 4: Delivery	0 days	Fri 1/13/06	Fri 1/13/06	14
16		Documentation	90 days	Sun 10/16/05	Fri 1/13/06	

Figure 1.2 Estimated work schedule

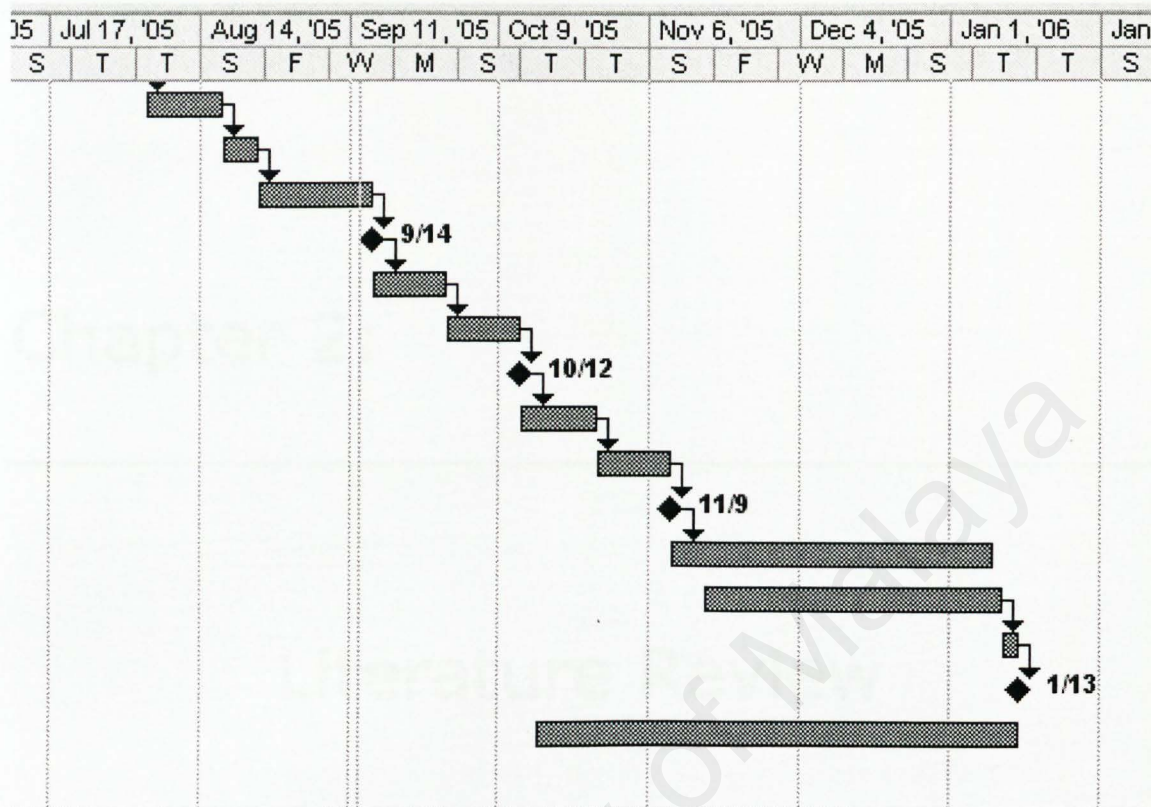


Figure 1.3 Gantt chart for project timeline

1.9 Summary

This chapter gives an overall review including the purposes and expectations of the project. It gives a general idea on how the end result should have and it takes into account of the existing methodologies being used. The project timeline is to make sure that all the tasks will be completed in time.

Chapter 2:

Literature Review

Chapter 2: Literature Review

2.1 Background study

Database technology can be considered the most vital in computer systems today and its applications range from single user to large database systems with terabytes of data supporting thousands of users. To manipulate the database, the Database Management System (DBMS) and Relational Database Management System (RDBMS) are introduced. Tools which will be discussed are used to give the RDBMS a new look and interface to ease and help users to operate on the RDBMS.

2.2 Introduction to database

A database is a collection of non-redundant data and it is used to separate the program and data by storing data in the database in another physical storage away from the application program itself. Database can reduce data redundancy and inconsistency while in the same time, helps to maintain the quality and integrity of the data. The database can present different view of data to users and this is controlled by the database administrator.

2.3 Database Management System (DBMS)

A database could not stand by itself and it must be managed, organized, and administered by a structure which gives meaning to the database and to hold its valuable data. To build this structure, a tool is needed and it is called, the Database Management System

(DBMS). In another word, the DBMS is a set of programs to process databases and their associated applications.

The reasons for having the DBMS is to reduce the amount of redundancy in the stored data, avoid inconsistency, and the most important is to provide a centralized control of the operational data.

Users communicate with the database through this Database Management System DBMS and besides acting as a medium to communicate with the database, DBMS has other usabilities. It provides the user with efficient access and in the same time, it takes care of data integrity and security. It allows user to access the database concurrently and be able to recover it when crashes happen.

2.4 Relational Database Management System (RDBMS)

Relational database came out of research at IBM and the University of California at Berkeley in the 1970s (Seltzer, 2005) and it's defined as a system whose users view data as a collection of tables related to each other through common data values (Kline, 2001). Data is stored in a 2 dimensional tables and multiple relationships between data elements can be defined and established in an ad-hoc manner.

To manage relational database, the Relational Database Management System is used and it is a system with the capability of recombining data elements to form different relations resulting in a great flexibility of data usage. Its advantages includes its fast retrieval speed for multi-user or transactional environment, new tables and rows can be added and modified easily and its flexibility. Its disadvantages are more on time and cost because it is easy to create an inefficient and badly database design if there is no proper data analysis prior to implementation. According to E.F. Codd's Twelve Principles of Relational Databases, a database product must meet all these twelve rules in order to be labeled as "relational" database product (Kline, 2001). The rules are:

- Information is represented logically in tables.
- Data must be logically accessible by table, primary key, and column
- Null values must be uniformly treated as "missing information," not as empty strings, blanks, or zeros.
- Metadata (data about the database) must be stored in the database just as regular data is.
- A single language must be able to define data, views, integrity constraints, authorization, transactions, and data manipulation.
- Views must show the updates of their base tables and vice versa
- A single operation must be able to retrieve, insert, update, or delete data.
- Batch and end-user operations are logically separate from physical storage and access methods.
- Batch and end-user operations can change the database schema without having to recreate it or the applications built upon it.

- Integrity constraints must be available and stored in the RDB metadata, not in an application program.
- The data manipulation language of the relational system should not care where or how the physical data is distributed and should not require alteration if the physical data is centralized or distributed.
- Any row processing done in the system must obey the same integrity rules and constraints that set-processing operations do.

A few examples of RDBMS are Oracle, MS SQL Server, PostgreSQL, Informix and MySQL, each with its own advantages and drawbacks.

2.4.1 Comparison between RDBMS

(Giacomo, 2005)

	MySQL	PostgreSQL	Oracle	MSSQL
Data storage				
Storage models	MyISAM, InnoDB, Berkeley DB, full-text High/very high Large/very large	Postgres	Bitmapped, B-tree, IOT, function-based	Clustered, nonclustered
Reliability		High	High/very high	High/very high
Scalability		Large	Large/very large	Large/very large
Indexes				
Single and multicolumn, primary key, and full text	Yes	Yes	Yes	Yes
Data integrity				
ACID compliance, row-level locking, hot backup, partial rollback	Yes	Yes	Yes	Yes
Replication				

Single master	Yes	Yes	Yes	Yes
Multimaster	Yes	Yes/no*	Yes	Yes
Clustering	Yes	No	Yes	Yes
Interface methods				
ODBC/JDBC, C/C++ AND Java	Yes	Yes	Yes	Yes
Advanced features				
Stored procedures, views, triggers, sequences and cursors	Yes (starting with version 5.x)	Yes	Yes	Yes

*solution exist but they are commercial

Table 2.1 Comparison between RDBMS

2.5 Structured Query Language (SQL)

SQL has been accepted as the standard query language for relational database system. It allows users to access data in the relational database system as well as to define and manipulate the data.

2.5.1 SQL standard

The first SQL standard was published in 1986 by the American National Standards Institute (ANSI) and the second widely adopted standard comes in 1989 (Suehring, 2002). ANSI then released an update in 1992 which is known as SQL1992 and in 1999, the SQL99 was borne. New features, commands and capabilities will be added in each releases and the International Standards Organization (ISO) has also approved the SQL99. The most recent release is the SQL:2003 which has replaced the SQL99. It makes some revisions and slight reorganization to some parts of the SQL99 while adding a brand new part which is the SQL/XML, with some XML-related specifications (Eisenberg, 2004).

2.5.2 SQL commands reference

The list below (Kline, 2001) shows the SQL commands in alphabetical order along with its statement classes and implementations based on the SQL99. Four vendors which are the MS SQL Server, MySQL, Oracle and PostgreSQL will be compared according to their vendor's level of support which can be divided into four categories:

- *Supported (S)*

The vendor supports the SQL99 standard for the particular command.

- *Supported, with variations (SWV)*

The vendor supports the SQL99 standard for the particular command, using vendor specific code or syntax.

- *Supported, with limitations (SWL)*

The vendor supports some but not all of the functions specified by the SQL99 standard for the particular command.

- *Not supported (NS)*

The vendor does not support the particular command according to the SQL99 standard.

Table 2.2 SQL commands reference

Command	SQL Statement Class	SQL99	MS SQL Server	MySQL	Oracle	Postgre SQL
ALTER PROCEDURE	SQL schema	Yes	SWV	NS	SWV	NS
ALTER TABLE	SQL schema	Yes	SWV	SWL	SWV	SWV
ALTER TRIGGER	SQL schema	No	SWV	NS	SWV	NS
ALTER VIEW	SQL schema	No	SWV	NS	SWV	NS

CALL	SQL control	Yes	NS	NS	S	S
CASE	SQL data	Yes	S	S	NS	S
CAST	SQL data	Yes	S	NS	NS	S
CLOSE CURSOR	SQL data	Yes	S	NS	S	S
COMMIT TRANSACTION	SQL transact- tion	Yes	SWV	NS	S	S
CONCAT- ENATION OPERATORS	SQL data	Yes	SWV	SWV	S	S
CONNECT	SQL transact- tion	Yes	SWL	NS	S	NS
CREATE DATABASE	SQL schema	No	SWV	S	S	SWV
CREATE FUNCTION	SQL schema	Yes	SWV	SWV	SWV	SWV
CREATE INDEX	SQL schema	Yes	SWV	SWV	SWV	SWV
CREATE PROCED- URE	SQL schema	Yes	S	NS	S	NS
CREATE ROLE	SQL schema	Yes	NS	NS	SWV	NS

CREATE SCHEMA	SQL schema	Yes	S	NS	S	NS
CREATE TABLE	SQL schema	Yes	SWV	SWV	SWV	SWV
CREATE TRIGGER	SQL schema	Yes	SWV	NS	SWV	SWV
CREATE VIEW	SQL schema	Yes	SWV	NS	SWV	SWV
DECLARE CURSOR	SQL data	Yes	S	NS	S	S
DELETE	SQL data	Yes	SWV	SWV	S	S
DISCON- NECT	SQL connec- tion	Yes	SWL	NS	SWV	NS
DROP DATABASE	SQL schema	Yes	SWV	SWV	NS	SWV
DROP FUNCTION	SQL schema	Yes	SWV	SWV	SWV	SWV
DROP INDEX	SQL schema	Yes	SWV	SWV	SWV	SWV
DROP PROCEDURE	SQL schema	Yes	S	NS	S	NS

DROP ROLE	SQL schema	Yes	NS	NS	SWV	NS
DROP TABLE	SQL schema	Yes	SWV	SWV	SWV	SWV
DROP TRIGGER	SQL schema	Yes	SWV	NS	SWV	SWV
DROP VIEW	SQL schema	Yes	S	NS	S	S
FETCH	SQL data	Yes	S	NS	S	SWV
GRANT	SQL schema	Yes	SWV	SWV	SWV	SWV
INSERT	SQL schema	Yes	SWV	SWV	S	S
JOIN clause	SQL data	Yes	S	SWL	NS(theta joins supported)	SWV (theta joins support- ed)
LIKE operator	SQL schema	Yes	SWV	SWV	SWV	SWV
OPEN	SQL schema	Yes	S	NS	S	S

OPERATORS	SQL schema	Yes	SWV	SWV	SWV	SWV
RETURN	SQL control	Yes	S	S	S	S
REVOKE	SQL schema	Yes	SWV	SWV	SWV	SWV
ROLLBACK	SQL transact- tion	Yes	SWV	NS	S	S
SAVEPOINT	SQL transact- tion	Yes	SWV	NS	S	NS
SELECT	SQL data	Yes	SWV	SWV	SWV	SWV
SET CONNECT- ION	SQL connec- tion	Yes	SWL	NS	NS	NS
SET ROLE	SQL session	Yes	NS	NS	SWV	NS
SET TIME ZONE	SQL session	Yes	NS	NS	SWV	NS
SET TRANSAC- TION	SQL session	Yes	SWV	NS	SWL	S
START TRANSAC- TION	SQL transact- tion	Yes	NS (support BEGIN TRAN)	NS	NS	NS(sup- ports BEGIN TRAN)

TRUNCATE TABLE	SQL data	Yes	S	NS	SWV	S
UPDATE	SQL data	Yes	SWV	SWV	SWV	S

2.6 MySQL

MySQL is owned by the company MySQL AB, a Sweden company and “mSQL” was used initially to connect database tables using fast low-level (ISAM) routines. But after some testing, the developers at MySQL AB found that mSQL was not fast and flexible enough. A new SQL interface to the database with almost the same API interface as mSQL was then introduced as a solution to the above problems and this is how MySQL was borne [5].

2.6.1 MySQL features

MySQL, which is arguably one of the most popular relational database management system (RDBMS) has its own attractive features which makes it stand equally with its competitors. Speed is one of its most accepted values as it is claimed to be one of the fastest database around. Other than this, MySQL offers clients to connect to the server at the same time and they may use multiple databases simultaneously, provided that it is supported by a variety of programming interfaces such as C, Perl, Java, PHP and Python (DuBois, 2003).

MySQL can be accessed from anywhere for data sharing since it is fully networked. In terms of security, MySQL has access control where viewing of data can be controlled and it now supports encrypted connections using the Secure Sockets Layer (SSL) protocol. MySQL runs on multi platform and since it comes with a small size, it can be distributed very easily.

In terms of scalability and limits, MySQL can handle large databases where it can handle a few million records. It now supports table up to 64 indexes and according to MySQL AB, MySQL 3.22 supports table up to 4 gigabyte. But with the MyISAM storage engine in MySQL 3.23 the maximum table size can be increased to 8 million terabytes [5].

2.6.2 MySQL datatypes

MySQL supports most of the SQL99 datatypes but not all of it and on the other hand, it also has several additional datatypes. The table below lists MySQL datatypes as a comparison to SQL99 datatypes (Kline, 2001).

Table 2.3 MySQL datatypes

MySQL Datatype	SQL99 Datatype	Description
bigint		Stores signed or unsigned integers within the range of -9223372036854775808 to 9223372036854775807.
char(n)[binary]	character(c)	Contains a fixed-length character string of 1 to 255

		characters in length, but trims spaces as <i>varchar</i> does. The BINARY option allows binary searches rather than dictionary-order, case-insensitive searches.
datetime	Datetime	Stores data and time values within the range of 1000-01-01 00:00:00 to 9999-12-31 23:59:59.
decimal	decimal(precision, scale)	Stores exact numeric values.
double(p,s), double precision	double precision	Holds double-precision numeric values
enum("val1," "val2," ... n)		Is a char datatype whose value must be one of those contained in the list of values. Up to 65535 distinct values are allowed.
float	float (p)	Stores floating-point numbers with a precision of or less.
int, integer	int, integer	Stores signed or unsigned integers within the range of -2147483548 to 2147483547.

longblob, longtext	binary large object	Stores BLOB or TEXT data up to 4294967295 characters in length.
mediumblob, mediumtext		Stores BLOB or TEXT data up to 65535 characters in length.
mediumint		Stores signed or unsigned integers within the range of -8388608 to 8388607.
nchar(n)[binary]	national character	Holds Unicode character strings, but is otherwise the same as char.
numeric(p,s)	numeric(p,s)	A synonym of decimal.
nvarchar(n) [binary]	nvarchar	Holds Unicode variable length character strings up to 255 characters in length.
real(p,s)	double precision	Is a synonym of double precision.
set("val1," "val2," ... n)		Is a char datatype whose value must be equal to zero or more values specified in the list of values. Up to 64

		items are allowed in the list of values.
smallint	smallint	Stores signed or unsigned integers within the range of –32758 to 32757.
timestamp(size)	timestamp	Stores the date and time within the range of 1970-01-01 00:00:00 to 2037-12-31 23:59:59.
tinyblob, tinytext		Is a BLOB or TEXT column of 255 characters or less.
tinyint		Stores signed or unsigned integers within the range of –128 to 127.
varchar(n)	character varying(n)	Stores variable-length character strings trimmed up to 255 characters in length.
year(2,4)		Stores either 2 or 4 year values, in the range of (19)70–(20)69 for 2-year format and 0000, 1901–

		2155 in 4-year format.
--	--	------------------------

2.6.3 MySQL supported functions

The table below shows the list of supported functions along with description of MySQL. Some of the functions are unique and it’s not guaranteed to work with other RDBMS [7].

Table 2.4 MySQL supported functions

Function	Description
<i>abs(X)</i>	Returns the absolute value of <i>X</i> .
<i>acos(X)</i>	Returns the arc cosine of <i>X</i> , i.e., the value whose cosine is <i>X</i> ; returns NULL if <i>X</i> is not in the range –1 to 1.
<i>ascii(str)</i>	Returns the ASCII code value of the leftmost character of the string <i>str</i> ; returns 0 if <i>str</i> is the empty string; returns NULL if <i>str</i> is NULL.
<i>asin(X)</i>	Returns the arc sine of <i>X</i> , i.e., the value whose sine is <i>X</i> ; returns NULL if <i>X</i> is not in the range –1 to 1.
<i>atan(X)</i>	Returns the arctangent of <i>X</i> , i.e., the value whose tangent is <i>X</i> .
<i>atan2(X,Y)</i>	Returns the arctangent of the two variables <i>X</i> and <i>Y</i> .
<i>avg(expr)</i>	Returns the average value of <i>expr</i> .
<i>benchmark(count,expr)</i>	Executes the expression <i>expr</i> <i>count</i> times. It may be used to time how fast MySQL processes the

	expression. The result value is always 0.
<i>binary</i>	Casts the string following it to a binary string.
<i>bin(N)</i>	Returns a string representation of the binary value of <i>N</i> , where <i>N</i> is a long (<i>BIGINT</i>) number.
<i>bit_count(N)</i>	Returns the number of bits that are set in the argument <i>N</i> .
<i>bit_and(expr)</i>	Returns the bitwise <i>AND</i> of all bits in <i>expr</i> . The calculation is performed with 64-bit (<i>BIGINT</i>) precision.
<i>bit_or(expr)</i>	Returns the bitwise <i>OR</i> of all bits in <i>expr</i> . The calculation is performed with 64-bit (<i>BIGINT</i>) precision.
<i>CASE value WHEN [compare-value] THEN result [WHEN [comparevalue] THEN result ...] [ELSE result] END</i> <i>CASE WHEN [condition] THEN result [WHEN [condition] THEN result ...] [ELSE result] END</i>	The first version returns the result where <i>value=comparevalue</i> . The second version returns the result for the first condition that is true. If there is no matching result value, then the result after <i>ELSE</i> is returned. If there is no <i>ELSE</i> part, NULL is returned.
<i>ceiling(X)</i>	Returns the smallest integer value not less than <i>X</i> .
<i>char(N,...)</i>	Interprets the arguments as integers and returns a string consisting of the characters given by the ASCII code values of those integers. NULL values are skipped.
<i>coalesce(list)</i>	Returns first non-NULL element in the list.
<i>concat(str1,str2,...)</i>	Returns the string that results from concatenating the arguments.
<i>concat_ws(separator, str1, str2,...)</i>	Stands for CONCAT With Separator and is a

	special form of <i>CONCAT()</i> . The first argument is the separator for the rest of the arguments. The separator and the rest of the arguments can be a string. If the separator is NULL, the result is NULL. The function skips any NULLs and empty strings after the separator argument. The separator is added between the strings to be concatenated.
<i>connection_id()</i>	Returns the connection ID (<i>thread_id</i>) for the connection. Every connection has its own unique ID.
<i>conv(N,from_base,to_base)</i>	Converts numbers between different number bases; returns a string representation of the number <i>N</i> , converted from base <i>from_base</i> to base <i>to_base</i> ; returns NULL if any argument is NULL.
<i>cos(X)</i>	Returns the cosine of <i>X</i> , where <i>X</i> is given in radians.
<i>cot(X)</i>	Returns the cotangent of <i>X</i> .
<i>count(DISTINCT expr,[expr...])</i>	Returns a count of the number of different values.
<i>count(expr)</i>	Returns a count of the number of non-NULL values in the rows retrieved by a <i>SELECT</i> statement.
<i>curdate()</i> <i>current_date</i>	Returns today's date as a value in 'YYYY-MM-DD' or YYYYMMDD format, depending on whether the function is used in a string or numeric context.
<i>curtime()</i> <i>current_time</i>	Returns the current time as a value in 'HH:MM:SS' or HHMMSS format, depending on whether the function is used in a string or numeric context.
<i>database()</i>	Returns the current database name.
<i>date_add (date,INTERVAL expr type)</i> <i>date_sub(date,INTERVAL expr</i>	These functions perform date arithmetic. <i>ADDDATE()</i> and <i>SUBDATE()</i> are synonyms for <i>DATE_ADD()</i> and <i>DATE_SUB()</i> . <i>date</i> is a

<i>type</i> <i>adddate(date,INTERVAL expr type)</i> <i>subdate(date,INTERVAL expr type)</i>	<i>DATETIME</i> or <i>DATE</i> value specifying the starting date. <i>expr</i> is an expression specifying the interval value to be added or subtracted from the starting date. <i>expr</i> may start with a - for negative intervals. <i>type</i> indicates how the expression should be interpreted.
<i>date_format (date,format)</i>	Formats the date value according to the format string.
<i>dayname(date)</i>	Returns the name of the weekday for date.
<i>dayofmonth(date)</i>	Returns the day of the month for date, in the range 1 to 31.
<i>dayofweek(date)</i>	Returns the weekday index for date (1 = Sunday, 2 = Monday, . . . 7 = Saturday).
<i>dayofyear(date)</i>	Returns the day of the year for date, in the range 1 to 366.
<i>decode(encrypt_str, pass_str)</i>	Decrypts the encrypted string <i>encrypt_str</i> using <i>pass_str</i> as the password. <i>encrypt_str</i> should be a string returned from <i>ENCODE()</i> .
<i>degrees(X)</i>	Returns the argument <i>X</i> , converted from radians to degrees.
<i>elt(N,str1,str2,str3,...)</i>	Returns <i>str1</i> if <i>N</i> = 1, <i>str2</i> if <i>N</i> = 2, and so on. Returns NULL if <i>N</i> is less than 1 or greater than the number of arguments. <i>ELT()</i> is the complement of <i>FIELD()</i> .
<i>encode(str,pass_str)</i>	Encrypts <i>str</i> using <i>pass_str</i> as the password. To decrypt the result, use <i>DECODE()</i> . The result is a binary string the same length as the string.
<i>encrypt(str[,salt])</i>	Encrypts <i>str</i> using the Unix <i>crypt()</i> system call. The <i>salt</i> argument should be a string with two characters.

<i>exp(X)</i>	Returns the value of e (the base of natural logarithms) raised to the power of X .
<i>export_set (bits,on,off,[separator, [number_of_bits]])</i>	Returns a string where every bit set in 'bit' gets an 'on' string and every reset bit gets an 'off' string. Each string is separated with 'separator' (default ',') and only 'number_of_bits' (default 64) of 'bits' is used.
<i>field(str,str1,str2,str3,...)</i>	Returns the index of <i>str</i> in the <i>str1, str2, str3, . . .</i> list. Returns 0 if <i>str</i> is not found. <i>FIELD()</i> is the complement of <i>ELT()</i> .
<i>find_in_set(str,strlist)</i>	Returns a value 1 to N if the string <i>str</i> is in the list <i>strlist</i> consisting of N substrings. A string list is a string composed of substrings separated by ',' characters. Returns 0 if <i>str</i> is not in <i>strlist</i> or if <i>strlist</i> is the empty string. Returns NULL if either argument is NULL. This function does not work properly if the first argument contains a ','.
<i>floor(X)</i>	Returns the largest integer value not greater than X .
<i>format(X,D)</i>	Formats the number X to a format like '#,###,###.##', rounded to D decimals. If D is 0, the result has no decimal point or fractional part.
<i>from_days(N)</i>	Given a daynumber N , returns a <i>DATE</i> value. Not intended for use with values that precede the advent of the Gregorian calendar (1582), due to the days lost when the calendar was changed.
<i>from_unixtime(unix_timestamp)</i>	Returns a representation of the <i>unix_timestamp</i> argument as a value in 'YYYY-MM-DD HH:MM:SS' or YYYYMMDDHHMMSS format, depending on whether the function is used in a string or numeric context.
<i>from_unixtime(unix_timestamp,for</i>	Returns a string representation of the

<i>mat</i>)	<i>unix_timestamp</i> , formatted according to the format string. Format may contain the same specifiers as those listed in the entry for the <i>DATE_FORMAT()</i> function.
<i>get_lock(str,timeout)</i>	Tries to obtain a lock with a name given by the string <i>str</i> , with a timeout of <i>timeout</i> seconds. Returns 1 if the lock is obtained successfully, 0 if the attempt times out, or NULL if an error occurs.
<i>greatest(X,Y,...)</i>	Returns the largest (maximum-valued) argument.
<i>hex(N)</i>	Returns a string representation of the hexadecimal value of <i>N</i> , where <i>N</i> is a long (<i>BIGINT</i>) number. This is equivalent to <i>CONV(N,10,16)</i> . Returns NULL if <i>N</i> is NULL
<i>interval(N,N1,N2,N3,...)</i>	Returns 0 if $N < N1$, 1 if $N < N2$, and so on. All arguments are treated as integers. It is required that $N1 < N2 < N3 < \dots < Nn$ for this function to work correctly.
<i>hour(time)</i>	Returns the hour for time, in the range 0 to 23.
<i>if(expr1,expr2,expr3)</i>	If <i>expr1</i> is TRUE ($expr1 \diamond 0$ and $expr1 \diamond$ NULL), then <i>IF()</i> returns <i>expr2</i> , else it returns <i>expr3</i> . <i>IF()</i> returns a numeric or string value, depending on the context in which it is used.
<i>ifnull(expr1,expr2)</i>	If <i>expr1</i> is not NULL, <i>IFNULL()</i> returns <i>expr1</i> ; otherwise it returns <i>expr2</i> . <i>IFNULL()</i> returns a numeric or string value, depending on the context in which it is used.
<i>isnull(expr)</i>	If <i>expr</i> is NULL, <i>ISNULL()</i> returns 1; otherwise it returns 0.
<i>insert(str,pos,len,newstr)</i>	Returns the string <i>str</i> . The substring begins at position <i>pos</i> and is 10 characters long, replaced by the string <i>newstr</i> .

<i>instr(str,substr)</i>	Returns the position of the first occurrence of substring <i>substr</i> in string <i>str</i> .
<i>last_insert_id([expr])</i>	Returns the last automatically generated value that was inserted into an <i>AUTO_INCREMENT</i> column.
<i>lcase(str) lower(str)</i>	Returns the string <i>str</i> with all characters changed to lowercase according to the current character-set mapping (default is ISO-8859-1 Latin1).
<i>least(X,Y,...)</i>	With two or more arguments, returns the smallest (minimumvalued) argument.
<i>left(str,len)</i>	Returns the leftmost <i>len</i> characters from the string <i>str</i> .
<i>length(str)</i> <i>octet_length(str)char_length(str)character_length(str)</i>	These functions return the length of the string <i>str</i> .
<i>load_file(file_name)</i>	Reads the file and returns the file contents as a string. The file must be on the server, and the user must specify the full pathname to the file and have the file privilege.
<i>locate(substr,str) position(substr IN str)</i>	Returns the position of the first occurrence of substring <i>substr</i> in string <i>str</i> . Returns 0 if <i>substr</i> is not in <i>str</i> .
<i>locate(substr,str,pos)</i>	Returns the position of the first occurrence of substring <i>substr</i> in string <i>str</i> , starting at position <i>pos</i> ; returns 0 if <i>substr</i> is not in <i>str</i> .
<i>log(X)</i>	Returns the natural logarithm of <i>X</i> .
<i>log10(X)</i>	Returns the base-10 logarithm of <i>X</i> .
<i>lpad(str,len,padstr)</i>	Returns the string <i>str</i> , left-padded with the string <i>padstr</i> until <i>str</i> is 10 characters long.
<i>ltrim(str)</i>	Returns the string <i>str</i> with leading-space characters removed.

<i>make_set(bits,str1,str2, ...)</i>	Returns a set (a string containing substrings separated by ',' characters) consisting of the strings that have the corresponding bits in bit set. <i>str1</i> corresponds to bit 0, <i>str2</i> to bit 1, etc. NULL strings in <i>str1</i> , <i>str2</i> , ... are not appended to the result.
<i>md5(string)</i>	Calculates a MD5 <i>checksum</i> for the string. Value is returned as a 32-long hex number.
<i>min(expr) max(expr)</i>	Returns the minimum or maximum value of <i>expr</i> . <i>MIN()</i> and <i>MAX()</i> may take a string argument; in such cases they return the minimum or maximum string value.
<i>minute(time)</i>	Returns the minute for time, in the range 0 to 59
<i>mod(N,M)</i>	% Modulo (like the % operator in C); returns the remainder of <i>N</i> divided by <i>M</i> .
<i>month(date)</i>	Returns the month for date, in the range 1 to 12.
<i>monthname(date)</i>	Returns the name of the month for date.
<i>now() sysdate() current_timestamp</i>	Returns the current date and time as a value in 'YYYY-MMDD HH:MM:SS' or YYYYMMDDHHMMSS format, depending on whether the function is used in a string or numeric context.
<i>nullif(expr1,expr2)</i>	If <i>expr1 = expr2</i> is true, returns NULL; otherwise returns <i>expr1</i> .
<i>oct(N)</i>	Returns a string representation of the octal value of <i>N</i> , where <i>N</i> is a long number. This is equivalent to <i>CONV(N,10,8)</i> . Returns NULL if <i>N</i> is NULL.
<i>ord(str)</i>	If the leftmost character of the string <i>str</i> is a multibyte character, returns the code of multibyte character by returning the ASCII code value of the character in the format of: ((first byte ASCII code)*256+(second byte ASCII code))*256+third

	byte ASCII code...] If the leftmost character is not a multibyte character, returns the same value as the <i>ASCII()</i> function does.
<i>password(str)</i>	Calculates a password string from the plain-text password <i>str</i> . This is the function that is used for encrypting MySQL passwords for storage in the Password column of the user grant table.
<i>period_add(P,N)</i>	Adds <i>N</i> months to period <i>P</i> (in the format YYMM or YYYYMM). Returns a value in the format YYYYMM. Note that the period argument <i>P</i> is not a date value.
<i>period_diff(P1,P2)</i>	Returns the number of months between periods <i>P1</i> and <i>P2</i> . <i>P1</i> and <i>P2</i> should be in the format YYMM or YYYYMM. Note that the period arguments <i>P1</i> and <i>P2</i> are not date values.
<i>pi()</i>	Returns the value of π .
<i>pow(X,Y)</i> <i>power(X,Y)</i>	Returns the value of <i>X</i> raised to the power of <i>Y</i> .
<i>quarter(date)</i>	Returns the quarter of the year for date, in the range 1 to 4.
<i>radians(X)</i>	Returns the argument <i>X</i> , converted from degrees to radians.
<i>rand()</i>	Returns a random floating-point value in the range 0 to 1.0.
<i>rand(N)</i>	If an integer argument <i>N</i> is specified, it is used as the seed value.
<i>release_lock(str)</i>	Releases the lock named by the string <i>str</i> that was obtained with <i>GET_LOCK()</i> . Returns 1 if the lock is released, 0 if the lock isn't locked by this thread (in which case the lock is not released), and NULL if the named lock doesn't exist.

<i>repeat(str,count)</i>	Returns a string consisting of the string <i>str</i> repeated <i>count</i> times. If <i>count</i> ≤ 0 , returns an empty string. Returns NULL if <i>str</i> or <i>count</i> are NULL.
<i>replace(str,from_str,to_str)</i>	Returns the string <i>str</i> with all occurrences of the string <i>from_str</i> replaced by the string <i>to_str</i> .
<i>reverse(str)</i>	Returns the string <i>str</i> with the order of the characters reversed.
<i>right(str,ten)</i>	Returns the rightmost 10 characters from the string <i>str</i> .
<i>round(X)</i>	Returns the argument <i>X</i> , rounded to an integer.
<i>round(X,D)</i>	Returns the argument <i>X</i> , rounded to a number with <i>D</i> decimals. If <i>D</i> is 0, the result has no decimal point or fractional part.
<i>rpad(str,len,padstr)</i>	Returns the string <i>str</i> , right-padded with the string <i>padstr</i> until <i>str</i> is ten characters long.
<i>rtrim(str)</i>	Returns the string <i>str</i> with trailing space characters removed.
<i>sec_to_time(seconds)</i>	Returns the seconds argument, converted to hours, minutes, and seconds, as a value in 'HH:MM:SS' or HHMMSS format, depending on whether the function is used in a string or numeric context.
<i>second(time)</i>	Returns the second for time, in the range 0 to 59.
<i>sign(X)</i>	Returns the sign of the argument as -1, 0, or 1, depending on whether <i>X</i> is negative, zero, or positive.
<i>sin(X)</i>	Returns the sine of <i>X</i> , where <i>X</i> is given in radians.
<i>soundex(str)</i>	Returns a <i>soundex</i> string from <i>str</i> . Two strings that sound "about the same" should have identical <i>soundex</i> strings. A "standard" <i>soundex</i> string is four characters long,

	but the <i>SOUNDEX()</i> function returns an arbitrarily long string. A <i>SUBSTRING()</i> can be used on the result to get a “standard” <i>soundex</i> string. All non-alphanumeric characters are ignored in the given string. All international alphabetic characters outside the A–Z range are treated as vowels.
<i>space(N)</i>	Returns a string consisting of <i>N</i> space characters.
<i>sqrt(X)</i>	Returns the nonnegative square root of <i>X</i> .
<i>std(expr)</i> <i>stddev(expr)</i>	Returns the standard deviation of <i>expr</i> . The <i>STDDEV()</i> form of this function is provided for Oracle compatability.
<i>strcmp(expr1,expr2)</i>	<i>STRCMP()</i> returns 0 if the strings are the same, –1 if the first argument is smaller than the second according to the current sort order, and 1 otherwise.
<i>substring(str,pos,len)</i> <i>substring(str FROM pos FOR len)</i> <i>mid(str,pos,len)</i>	Returns a substring 10 characters long from string <i>str</i> , starting at position <i>pos</i> . The variant form that uses <i>FROM</i> is ANSI SQL92 syntax.
<i>substring_index (str,delim,count)</i>	Returns the substring from string <i>str</i> after <i>count</i> occurrences of the delimiter <i>delim</i> . If <i>count</i> is positive, everything to the left of the final delimiter (counting from the left) is returned. If <i>count</i> is negative, everything to the right of the final delimiter (counting from the right) is returned.
<i>substring(str,pos)</i> <i>substring(str FROM pos)</i>	Returns a substring from string <i>str</i> starting at position <i>pos</i> .
<i>sum(expr)</i>	Returns the sum of <i>expr</i> . Note that if the return set has no rows, it returns NULL.
<i>tan(X).</i>	Returns the tangent of <i>X</i> , where <i>X</i> is given in radians.
<i>time_format (time,format)</i>	This is used like <i>DATE_FORMAT()</i> , but the format string may contain only those format specifiers that

	handle hours, minutes, and seconds. Other specifiers produce a NULL value or 0.
<i>time_to_sec(time)</i>	Returns the time argument, converted to seconds.
<i>to_days(date)</i>	Given a date, returns a daynumber (the number of days since year 0).
<i>trim([[BOTH LEADING TRAILING] [remstr] FROM] str)</i>	Returns the string <i>str</i> with all <i>remstr</i> prefixes and/or suffixes removed. If none of the specifiers <i>BOTH</i> , <i>LEADING</i> , or <i>TRAILING</i> are given, <i>BOTH</i> is assumed. If <i>remstr</i> is not specified, spaces are removed.
<i>truncate(X,D)</i>	Returns the number <i>X</i> , truncated to <i>D</i> decimals. If <i>D</i> is 0, the result has no decimal point or fractional part.
<i>ucase(str) upper(str)</i>	Returns the string <i>str</i> with all characters changed to uppercase according to the current character set mapping (default is ISO-8859-1 Latin1).
<i>unix_timestamp()</i> <i>unix_timestamp(date)</i>	If called with no argument, returns a Unix timestamp (seconds since '1970-01-01 00:00:00' GMT). If <i>UNIX_TIMESTAMP()</i> is called with a date argument, it returns the value of the argument as seconds since '1970-01-01 00:00:00' GMT.
<i>user()</i> <i>system_user()</i> <i>session_user()</i>	These functions return the current MySQL username.
<i>version()</i>	Returns a string indicating the MySQL server version.
<i>week(date) week(date,first)</i>	With a single argument, returns the week for date, in the range 0 to 53. (The beginning of a week 53 is possible during some years.) The two-argument form of <i>WEEK()</i> allows the user to specify whether the week starts on Sunday (0) or Monday (1).
<i>weekday(date)</i>	Returns the weekday index for date (0 = Monday, 1

	= Tuesday, . . . 6 = Sunday).
<i>year(date)</i>	Returns the year for date, in the range 1000 to 9999.
<i>yearweek(date) yearweek(date,first)</i>	Returns year and week for a date. The second argument works exactly like the second argument to <i>WEEK()</i> . Note that the year may be different from the year in the date argument for the first and the last week of the year.

2.6.4 MySQL optimization

Optimization is a way to make the system runs faster and it requires understanding of the entire system to be optimized. Database is built by tables and sets and operations on these tables and sets involve queries. Therefore, ways to speed up the operations can be done by finding ways to speed up these queries. There are a number of ways to do this and the most common one is by the help of indexing because by indexing the tables, it may help the database server to look up rows more quickly. The ways queries are written will also affect the speed of the system to look up for the indexes.

The MySQL Query Optimizer takes advantage of the indexes while in the same time, it uses other information by eliminating rows which has nothing to do with the queries and narrow down the number of rows it has to search. It is best to help the Query Optimizer to take advantage of the indexes by using the following guidelines [9]:

- ***Compare columns that have the same type.*** Identical columns will give better speed performance than dissimilar types. For example, INT and BIGINT are different types while CHAR(10) and VARCHAR(10) are the same type.
- ***Try to make indexed columns stand alone in comparison expressions.*** It is better to write queries which lessen the number of times of the optimizer to retrieve rows from the database to do its comparison with an operation. The use of indexed columns should be maximized in the queries to find matching values in the columns.
- ***Don't use wildcards as a habit.*** The use of wildcards should be controlled and only be used when necessary.
- ***Test alternate forms of queries, but run them more than once.*** Run alternate forms of queries several times when testing the performance of the queries because disk cache stores past queries information and affects the speed of the test result. Queries should be run when the system is stable to increase the accuracy.
- ***Avoid overuse of MySQL's automatic type conversion.*** It is better to avoid automatic type conversions because the operation itself involves a small performance penalty.
- ***Don't use longer columns when shorter one is sufficient.*** For fixed length columns, it is a good practice to use the size of the column sufficiently. For example, if CHAR(40) is enough, avoid to declare CHAR(255).
- ***Use the appropriate row storage format for the table type.*** Fixed-length rows can be processed faster than variable-length rows although fixed-length row requires more space. For InnoDB tables, fixed-length row is better in terms of speed as the internal row storage format does not treat fixed-length row different than variable-length row.

- **Declare columns to be NOT NULL .** This can speed up processing and requires less storage because NULL values will not be checked
- **Use ENUM columns when necessary.** If a string column contains only a limited number of distinct values, changing it to an ENUM column will increase the processing speed as they are represented as numeric values internally
- **OPTIMIZE TABLE.** Tables especially those which contain variable-length columns are exposed to fragmentation where speed will be reduced after fragmentation because unused spaces will be used to store tables and this lead to performance reduce as more blocks need to be read to get the valid rows. OPTIMIZE TABLE helps to keep performance on the table from degrading.
- **Pack data into BLOB column.** Data which is stored using BLOB can be pack and unpack in application and can be retrieved in a single retrieval operation.
- **Use synthetic index.** Synthetic index can be created as a hash value based on other columns and store it in a separate column. Rows can then be found by searching the hash values but this method is only good for exact-match queries.
- **Make data and tables as small as possible.** This can be done by designing the size of the tables as little as possible. It gives a huge speed improvement as disk can be read faster while smaller table requires less main memory during queries execution.

2.7 Existing tools to manage MySQL

MySQL comes with the command line interface where users have to enter SQL command to manipulate the database. This is one of the disadvantages of using MySQL as users have to know the exact syntax to manipulate the database and it's not suitable for beginners who have weak knowledge on SQL.

```
mysql> use simple
Database changed
mysql> select * from test
-> ;
```

Column1	Column2	Column3	Column4	Column5
1	DummyValue	DummyValue	DummyValue	DummyValue
10	DummyValue	DummyValue	DummyValue	DummyValue
11	DummyValue	DummyValue	DummyValue	DummyValue
12	DummyValue	DummyValue	DummyValue	DummyValue
2	DummyValue	DummyValue	DummyValue	DummyValue
3	DummyValue	DummyValue	DummyValue	DummyValue
4	DummyValue	DummyValue	DummyValue	DummyValue
5	DummyValue	DummyValue	DummyValue	DummyValue
6	DummyValue	DummyValue	DummyValue	DummyValue
7	DummyValue	DummyValue	DummyValue	DummyValue
8	DummyValue	DummyValue	DummyValue	DummyValue
9	DummyValue	DummyValue	DummyValue	DummyValue

```
12 rows in set (0.00 sec)

mysql> _
```

Figure 2.1 MySQL command line interface

To overcome this, tools to manage MySQL database have been designed to ease users in terms of speed to manipulate the database and understanding of the whole structure. Most of these tools can be downloaded for free and examples of a few existing tools are:

2.7.1 Query Browser

MySQL Query Browser is a free tool which can be downloaded from the MySQL official website. It's one of the easiest tools to use where user can create, execute and optimize SQL queries for the database in its simple web browser like interface. Other than this, it gives the user flexibility on whether to use its interface or hand coding the query because the code for all queries and statements are displayed and can be easily edited or modified. By using the Object Browser, it allows us to choose the database and tables to query and browse through the previous issued queries to use them again. To visually create and modify tables and columns information, the Table Editor can be used. Other than this, the Script Editor provides an interface for creating, editing and debugging large SQL scripts which involves multiple SQL statements.

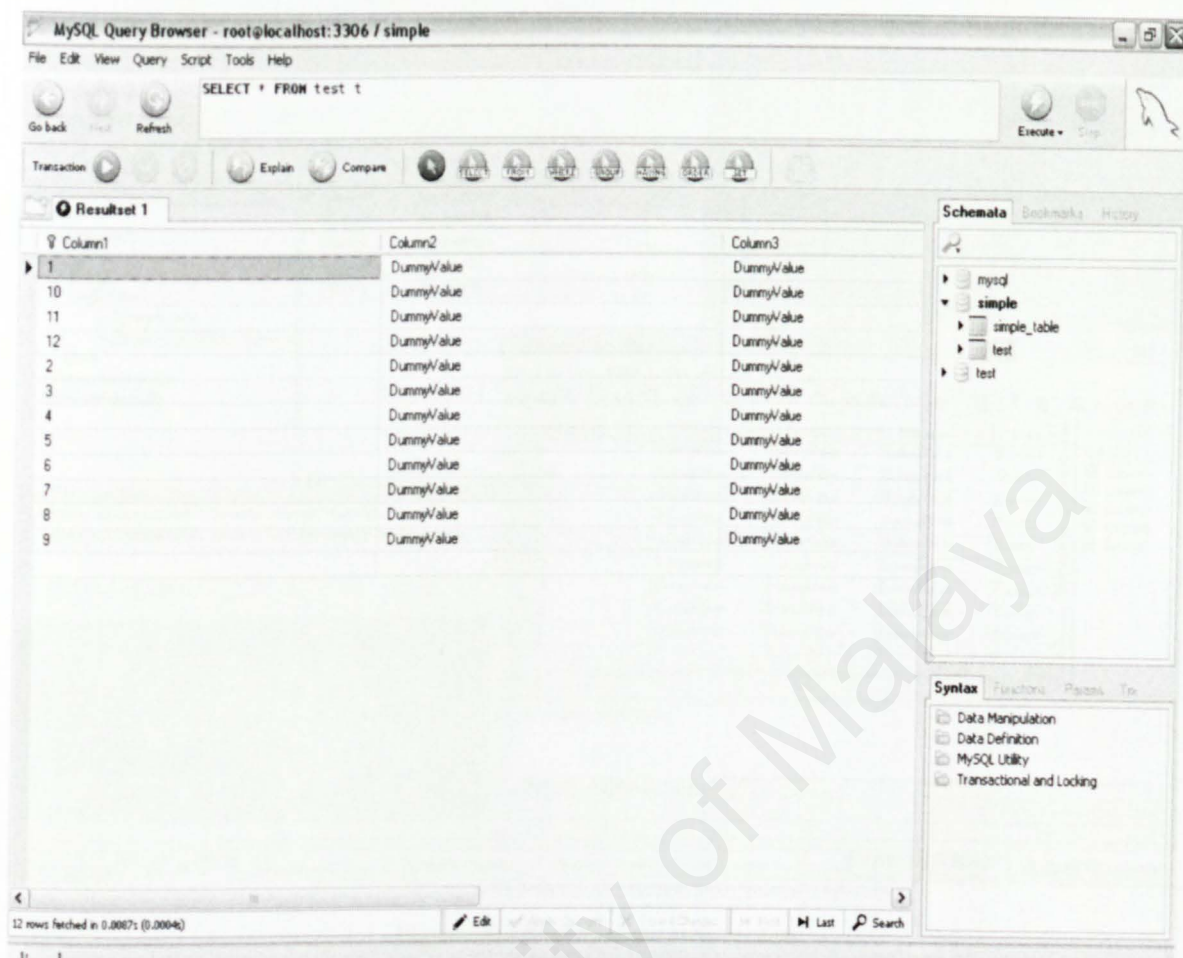


Figure 2.2 Query Browser

2.7.2 MySQL Control Center

MySQL Control Center provides interactive queries with a syntax highlighting SQL editor where user can construct queries and view the results in a configurable table display. It also provides an easy way to create and manage databases and tables including viewing, check, repair and optimize tables.

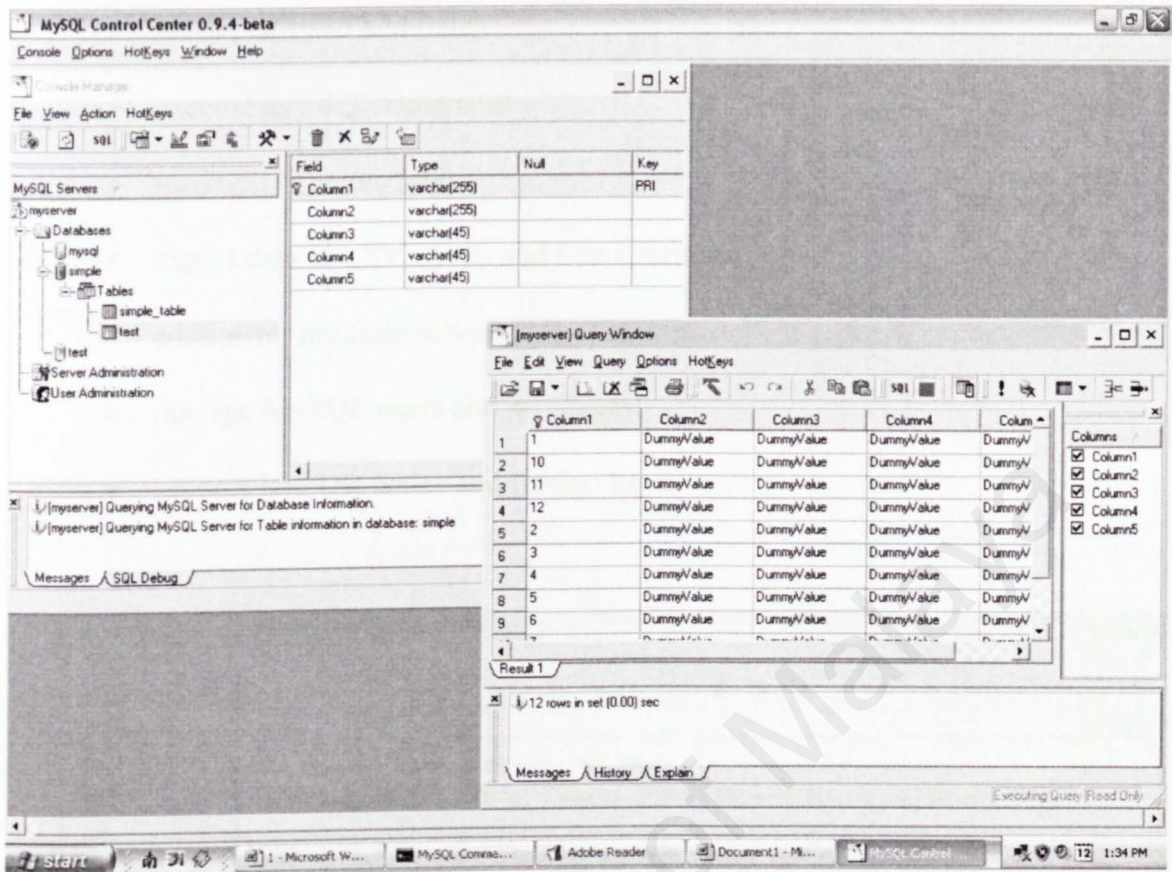


Figure 2.3 MySQL control center

2.7.3 phpMyAdmin

phpMyAdmin is a powerful tool which is able to manage the whole MySQL server and it can be run on many platforms including the Windows and Linux. What makes it so popular is its capability to manage the database via a web browser with the help of the web server. Some of its features are like [6]:

- creating and dropping databases
- create, copy, drop, rename and alter tables
- do table maintenance

- delete, edit and add fields
- execute any SQL statement
- load text files into tables
- export data to CSV, XML and Latex formats
- administer multiple servers
- manage MySQL users and privileges
- support InnoDB tables and foreign keys

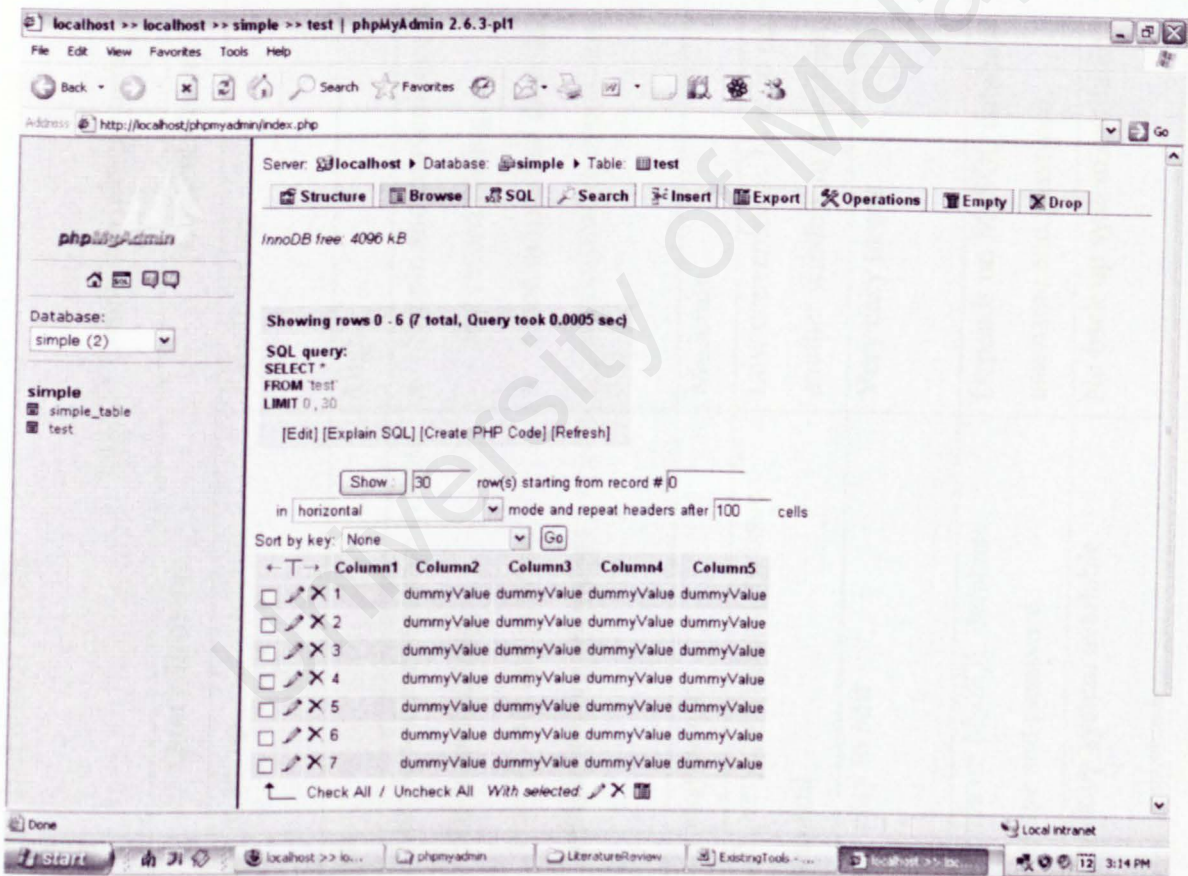


Figure 2.4 phpMyAdmin

2.7.4 Comparison between tools

Table 2.5 Comparison between tools

Features	Query Browser	Control Center	phpMyAdmin
Setting up	Easy to set up	Easy to set up	Not easy to set up, requires knowledge of PHP
Reliability	High	High	High
Built in features	<ul style="list-style-type: none">• Simple and convenience in table and record editing.• Poor in user and database administrative operations	<ul style="list-style-type: none">• Simple and convenience in table and record editing.• Good in user and database administrative operations.	<ul style="list-style-type: none">• Very rich in features• Be able to do operations with clickable buttons and links• Good in user and database administrative operations.
Ease of use	Very easy to use	Moderate	Quite complicated
SQL queries handiness	High, very convenience to issue command	Low convenience, have to open another window to issue command	Moderate. SQL command line interface can be accessed in tab
Record editing	Very easy to edit	Very easy to edit	Quite confusing and hard to edit due to complicating features
Access authentication	Depends on MySQL database username and password.	Depends on MySQL database username and password.	User may choose whether authentication is required
Back up	No back up system available	No back up system available	Back up system is available, but only in text file

2.7.5 MySQLadminPro features improvement

The MySQLadminPro will adopt other tools advantages and improve on their drawbacks

while in the same time, provides an easy to use environment. The main priority is its ease of use to increase efficiency and to overcome the MySQL command line interface difficulty. To improve on phpMyAdmin complex interface because of providing too much complicated information in the same page, MySQLadminPro will sort its operations carefully and systematically to provide a simple to use environment. Other complicated functions will be listed differently for advance users. To overcome the back up feature where other tools lack, MySQLadminPro will provide backup compressing feature too where the database can be backed up in a zip file format. The table below summarizes the features that MySQLadminPro can improve from other tools:

Table 2.6 MySQLadminPro features improvement

Features	MySQLadminPro
Setting up	Easy to set up
Reliability	High
Built in features	<ul style="list-style-type: none">• Simple and convenience in table and record editing.• Database administrative operations will be separated from database operations to avoid complexity
Ease of use	Very easy to use
SQL queries handiness	High, very convenience to issue command
Record editing	Very easy to edit
Access authentication	Depends on MySQL database username and password.
Back up	Compress database into zip file format

2.8 Summary

This chapter has given a brief introduction and background of database and Relational Database Management System (RDBMS). It also elaborated the Structured Query Language (SQL) commands, data types and functions in details. Existing tools to manage the RDBMS has been discussed and compared to understand their pros and cons so that improvements can be done.

Chapter 3:

System Analysis

Chapter 3: System Analysis

3.1 Introduction

The main purpose of system analysis is to gather deeper understanding on the requirements which may affect the resulting design and implementation later. System analysis will be 2 main categories which are the analysis of survey conducted and analysis of functional and non-functional requirements

3.2 Analysis of survey

The main purpose of this survey is to gather public opinions and views regarding relational database management system (RDBMS) and how would they want a tool to help them in manipulating the MySQL database. By reviewing the result of this survey, we can better understand what users want and their general needs.

A total of 30 undergraduates from FSKTM which has experienced the MySQL database before were chosen to conduct this survey using the survey form (Appendix A).

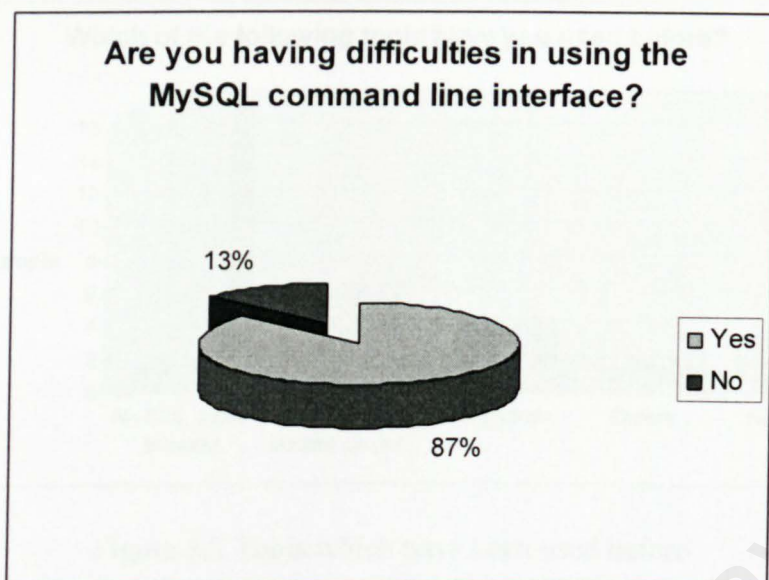


Figure 3.1 Difficulties in using MySQL command line interface

Majority of the users of MySQL (around 87 %) is having difficulties in using the command line interface. This is probably due to the difficulty in memorizing all the SQL syntaxes and uses them without error since from the survey, 60% of the respondents claimed that they are poor in SQL language, 30% claimed that they are moderate and only around 10% claimed that they are good in the language.

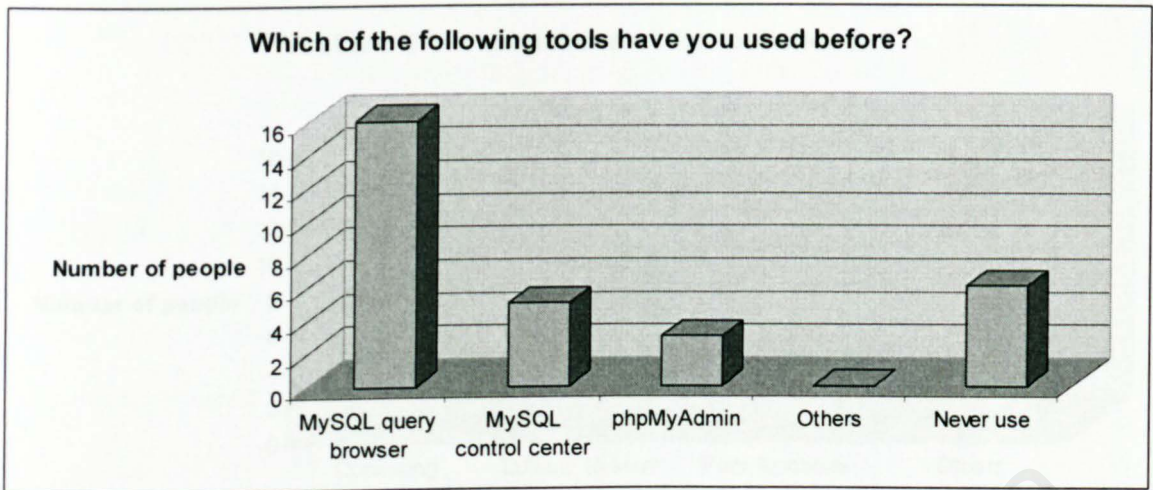


Figure 3.2 Tools which have been used before

Among all MySQL users, they prefer the MySQL query browser, most probably because it can be easily downloaded from the MySQL official website. This is followed by MySQL control center and phpMyAdmin. The number of respondents who never use any tool during operation with MySQL is quite many too as a comparison.

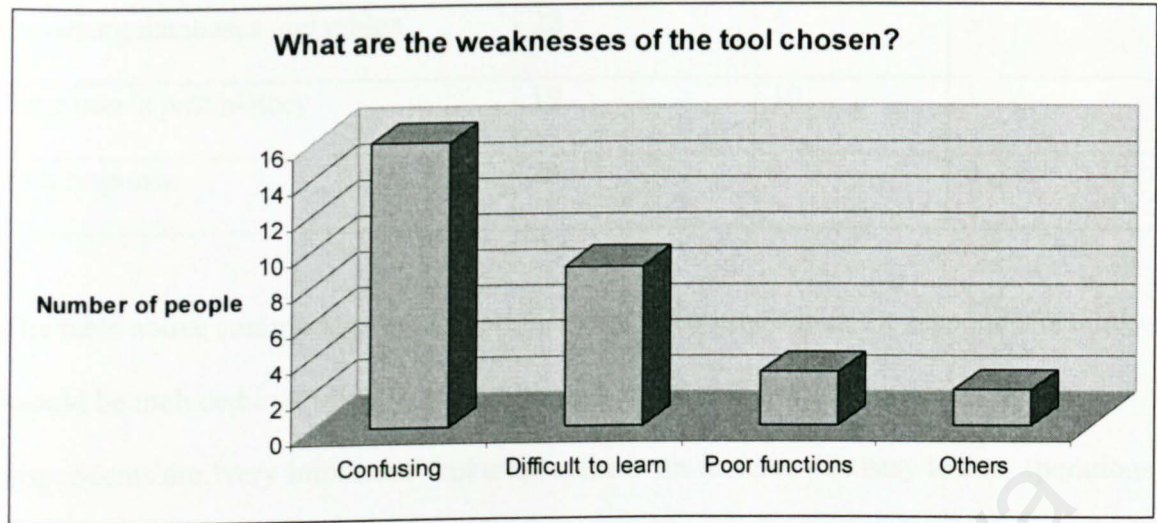


Figure 3.3 Weaknesses of the tool chosen

The main weaknesses of the tools which have been used by the respondents are confusing. This may be due to complicated interface design and functions. Other than that, difficult to learn and poor functions are two other drawbacks. Other weaknesses mentioned by the respondents are like difficult to set up the tool and not user friendly

Table 3.1 Importance of the features of MySQL tool

Features	Importance (Number of selection)		
	Very important	Moderate	Not important
Easy to use	23	7	0
Lots of functions	16	10	4
Adding, deleting and editing records	28	2	0
Database compression	15	9	6

Browsing databases and tables	22	5	3
Save user's past history	19	10	1
Fast response	29	1	0

The table above summarizes the importance of the features which the respondents think should be included in a MySQL tool. Although most of the features selected by the respondents are ‘very important’ but emphasize is on features like easy to use, operations on records, database and tables browsing and fast response.

3.3 Requirements

Requirements are defined during the early stages of a system development as a specification of what should be implemented [12]. Besides, it is important to ensure that a clear understanding between the developer and client is acquired. Requirements for the MySQLAdminPro can be divided into four categories which are functional requirements, non-functional requirements, hardware, and software requirements.

3.3.1 Functional Requirements

Functional requirements are necessities which specify actions or results that the intended system supposes to deliver. The functional requirements for MySQLAdminPro can be described using the use case diagram below:

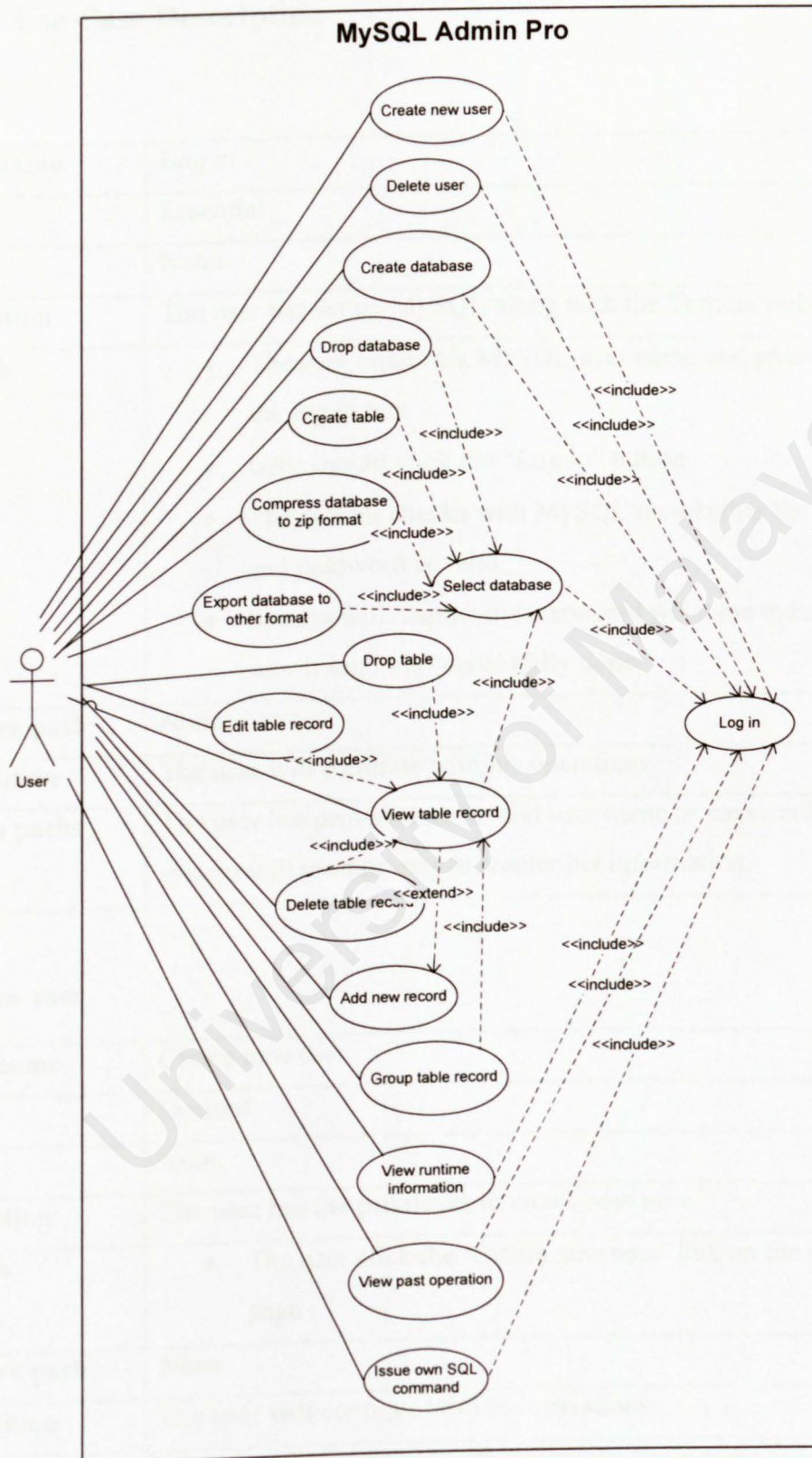


Figure 3.4 Use case diagram

3.3.1.1 Use Case Description

Log in

Use case name	Log in
Priority	Essential
Trigger	None
Pre-condition	The user has set up MySQL along with the Tomcat web server
Basic path	<ul style="list-style-type: none">• The user inputs his MySQL user name and password into the log in box• User should click the “Log in” button• The system checks with MySQL on whether the username and password is valid• System will display the menu and welcome message to the user if log in is successfully done
Alternative path	None
Post condition	The user will continue with his operations
Exception paths	The user has provided an invalid user name or password, and the system will prompt user to reenter his information

Create new user

Use case name	Create new user
Priority	Desired
Trigger	None
Pre-condition	The user has the privileges to create new user
Basic path	<ul style="list-style-type: none">• The user click the ‘Create new user’ link on the menu page
Alternative path	None
Post condition	The user will continue with his operations
Exception paths	The user will be prompt with a message stating that the user could not be created if the user does not have the right privileges

Delete user

Use case name	Delete user
Priority	Desired
Trigger	None
Pre-condition	The user has the privileges to delete user
Basic path	<ul style="list-style-type: none">The user click the 'View all user' link on the menu page and select the user to be deleted
Alternative path	None
Post condition	The user will continue with his operations
Exception paths	The user will be prompt with a message stating that the user could not be deleted if the user does not have the right privileges

Select database

Use case name	Select database
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system
Basic path	<ul style="list-style-type: none">The user choose the available database from the "select database" drop down list
Alternative path	None
Post condition	All tables which contains in the selected database will be displayed to the user
Exception paths	None

Create new database

Use case name	Create new database
Priority	Desired

Trigger	None
Pre-condition	The user has successfully log into the system
Basic path	<ul style="list-style-type: none"> • The user enters the database name in the “Create new database” text box • The user then should click on the “Create” button
Alternative path	None
Post condition	A message stating the database has successfully created will be displayed
Exception paths	The user will be prompt with a message stating that an error has occurred if an invalid database name is entered

View all database

Use case name	View all database
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system
Basic path	<ul style="list-style-type: none"> • The user click the “View all database” link on the menu page
Alternative path	None
Post condition	All available database in MySQL will be displayed
Exception paths	None

Select table

Use case name	Select table
Priority	Desired
Trigger	None

Pre-condition	The user has successfully log into the system and has chosen a database
Basic path	<ul style="list-style-type: none"> • The user should select a database • The user should click on the preferred table from the table list
Alternative path	None
Post condition	All records in the table will be displayed to the user
Exception paths	None

Create new table

Use case name	Create new table
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system and has chosen a database
Basic path	<ul style="list-style-type: none"> • User should select a database • The user should enter the preferred table name • The user should enter the number of columns for the new table • They user should click on the “Create new table” button • User has to enter the table fields and select the data type for the fields • A primary key should be selected • When all necessary data has been entered, user should click on the “Submit” button
Alternative path	None
Post condition	All the tables including the new created one will be displayed to the user

Exception paths	A create table error message will be displayed to the user if invalid data is entered
------------------------	---

Drop database

Use case name	Drop database
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system
Basic path	<ul style="list-style-type: none"> • The user must select the “View all databases” link from the menu page • User should select the preferred database to be dropped by selecting the checkbox beside the database • User should click the “Drop database” button
Alternative path	None
Post condition	All database will be displayed to the user and a message stating that the database has been dropped will be displayed
Exception paths	An error message will be displayed if the user did not select any database before pressing the “Drop database” button

Drop table

Use case name	Drop table
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system and chosen a database
Basic path	<ul style="list-style-type: none"> • The user choose the preferred database

	<ul style="list-style-type: none"> • User should select the preferred table to be dropped by selecting the checkbox beside the table • User should click the “Drop table” button
Alternative path	None
Post condition	All tables will be displayed to the user and a message stating that the table has been dropped will be displayed
Exception paths	An error message will be displayed if the user did not select any table before pressing the “Drop table” button

Insert new record

Use case name	Insert new record
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system and chosen a database and a table
Basic path	<ul style="list-style-type: none"> • The user choose the preferred database and table • User should click on the “Insert new record” button • User should then enter the values for all the necessary fields of the table • User should then click on the “Submit” button
Alternative path	None
Post condition	All records will be displayed to the user including the newly created one and a message stating that the record is successfully created will be displayed
Exception paths	An error message will be displayed if the user did not enter the correct data type for the table field

Edit record

Use case name	Edit record
Priority	Desired
Trigger	None
Pre-condition	The user has successfully log into the system and chosen a database and table
Basic path	<ul style="list-style-type: none">• The user chooses the preferred database and table• The user click on the “Edit” button besides the record displayed• The user enters all the preferred new values to replace the current values for the record• User then should click on the “Submit” button
Alternative path	None
Post condition	All records will be displayed to the user and a message stating that the record has been added will be displayed
Exception paths	An error message will be displayed if the user did not enter a valid data type for the record

Delete record

Use case name	Delete record
Priority	Desired
Trigger	None
Pre-condition	<ul style="list-style-type: none">• The user has successfully log into the system and chosen a database and table• The user has the privilege to delete record
Basic path	<ul style="list-style-type: none">• The user chooses the preferred database and table• The user select the record to be deleted• User then should click on the ‘Drop record’ button
Alternative path	None

Post condition	All records will be displayed to the user and a message stating that the record has been deleted will be displayed
Exception paths	An error message will be displayed if the user did not select a record to delete

Export database to other format

Use case name	Export database to other format
Priority	Desired
Trigger	None
Pre-condition	<ul style="list-style-type: none"> The user has successfully log into the system
Basic path	<ul style="list-style-type: none"> The user click the 'Export' link on the menu The user select the database to be exported The user select the format to be exported The user select the table to be exported
Alternative path	None
Post condition	A message will be displayed stating that the selected table has been exported
Exception paths	An error message will be displayed if the user did not select a database or a table or the right file format to be exported

Compress database to zip format

Use case name	Compress database to zip format
Priority	Desired
Trigger	None
Pre-condition	<ul style="list-style-type: none"> The user has successfully log into the system
Basic path	<ul style="list-style-type: none"> The user click the 'Backup' link on the menu The user select the database to be compressed The user select the table to be compressed
Alternative path	None

Post condition	A message will be displayed stating that the selected table has been compressed
Exception paths	An error message will be displayed if the user did not select a database or a table to be compressed

View past operation

Use case name	View past operation
Priority	Desired
Trigger	None
Pre-condition	<ul style="list-style-type: none"> The user has successfully log into the system
Basic path	<ul style="list-style-type: none"> The user click the ‘Show user log’ link on the menu
Alternative path	None
Post condition	A file containing all the user’s past operation will be open
Exception paths	No file will be opened

3.3.2 Non-functional Requirements

Non functional requirements define the overall quality or attributes of the resulting system. They are not specifically concerned with the functionality of a system but they place restrictions on the product being developed, the developing process and they also specify the external constraints that the product must meet [11].

- Usability**

The system shall provide an easy to use and not confusing environment for the user to improve efficiency so that they may benefit the most from the system.

Functions and features are arranged and formatted well so that user may reach them straightforwardly

- **Security**

The system provides security where only valid MySQL user is permitted to enter the system. All access and privileges of an user in the system is the same with the MySQL database to ensure access right and to data protection

- **Availability**

The system is available to the user at all time after all the setup has been done.

- **Maintainability**

The system is easy to maintain as all the codes written are open to be modified and edited to correct faults and bugs.

- **Reliability**

The system should be well-tested so that all functions are proven reliable to avoid costly downtime and incorrect results delivered

3.3.3 Hardware Requirements

Below is the basic hardware specification which is needed to run the system:

Table 3.2 Hardware requirements

Hardware	Requirements
Processor	Pentium III 800 Mhz
Memory (Ram)	256 MB
Hard disk space	200 MB (Depends on database size)
Others	Standard computer hardware

3.3.4 Software Requirements

Below is the basic software needed to run the system:

Table 3.3 Software requirements

Software	Requirements
Operating system	Microsoft Windows
Web Server	Apache Tomcat
Database	MySQL
Browser	Microsoft Internet Explorer

Chapter 4:

System Design

Chapter 4: System Design

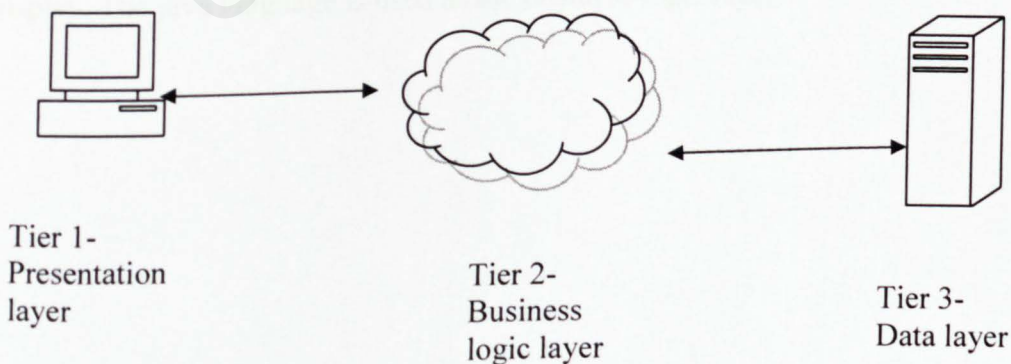
4.1 Introduction

System design will show how actually the whole system is going to achieve what it is intended to do and whether it meets the objectives and requirements specified. In another words, it can be called physical design which can be divided into 4 stages as below:

- Architecture design
- Process design
- Input and output design
- Interface design

4.2 Architecture design

MySQLAdminPro is built as a 3-tier architecture which consists of a presentation layer as the first tier, the business logic as the second tier and lastly the data layer as the third tier:



4.2.1 The Presentation Layer

User will interact with the system from this tier which comprise of a browser as a medium of interaction. Interactive pages will be displayed to answer requests from the user and it acts as a communication bridge between the user and the business logic layer. In another hand, information is sent from the back-end to the user and will be displayed in this layer too. The presentation layer will be formatted in JavaServer Pages (JSP) using HTML to be displayed on a browser along with cascading style sheets (CSS) to make it look more appealing

4.2.2 The Business Logic Layer

All requests and interactions of the user from the presentation layer will be sent to the business layer to process. From the user's requests, the business logic layer will determine what actions should be taken and in the same time, implements it. This layer acts as a communication bridge between the presentation and the data layer which means it will also handles results which is sent back from the data layer due to user's requests. The results will then be processed accordingly before being sent to the presentation layer for display. The java language is used as the business logic layer.

4.2.3 The Data Layer

The data layer primary function is to provide fast and reliable access to the data needed to run the whole system efficiently. Since MySQLAdminPro is a database administer system which helps user to query and analyze data stored in the MySQL database, it is important to ensure data manipulation process like data creation, retrieving, deletion and updates can be done effectively.

4.3 Process Design

The structure and the flow of data throughout the whole system will be monitored and organized in the process design. A few core sequence diagrams and data flow diagrams (DFD) will be used here to elaborate the process activities:

4.3.1 Context diagram

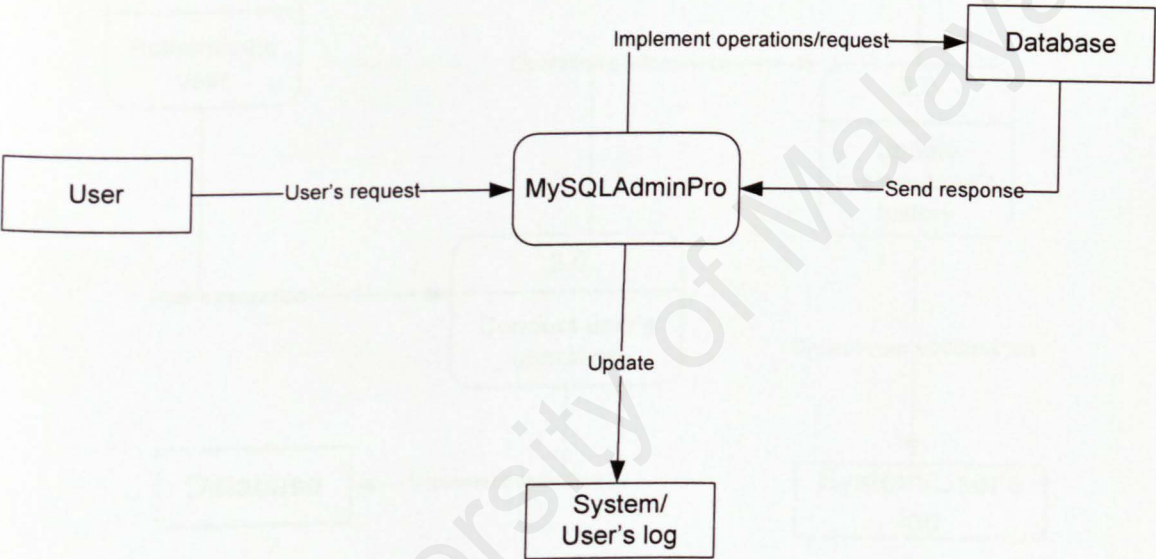


Figure 4.1 Context diagram

4.3.2 Level 0 DFD

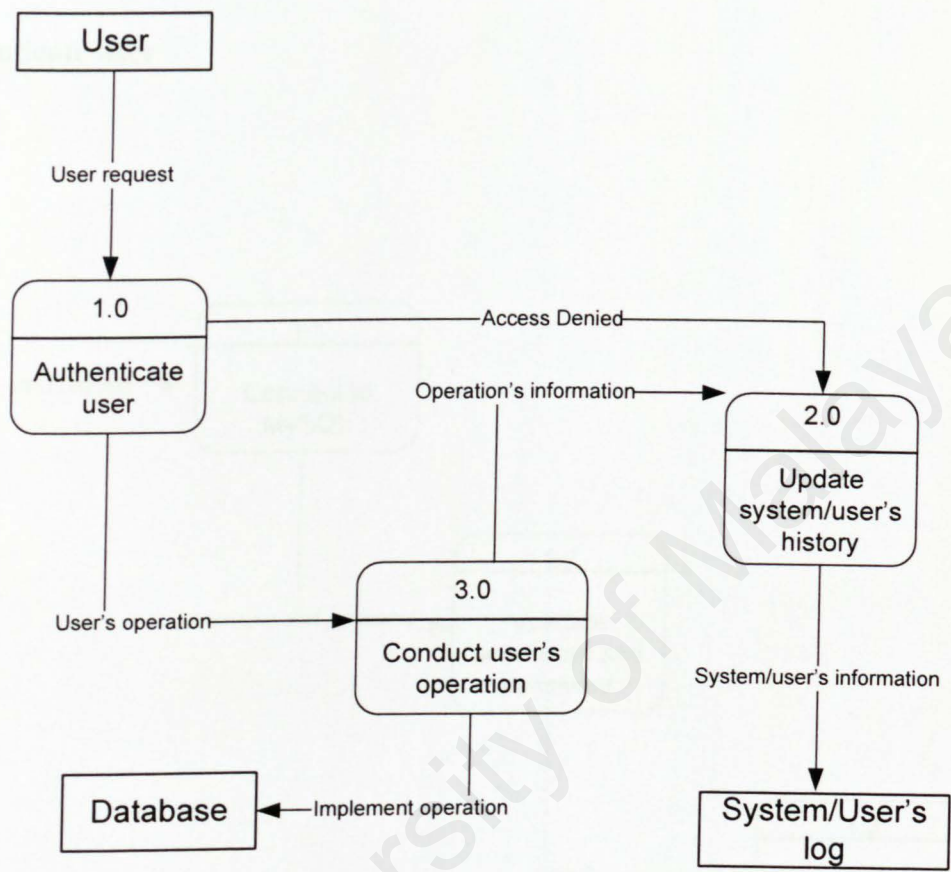


Figure 4.2 Level 0 DFD

4.3.3 Level 1 DFD

Authenticate user

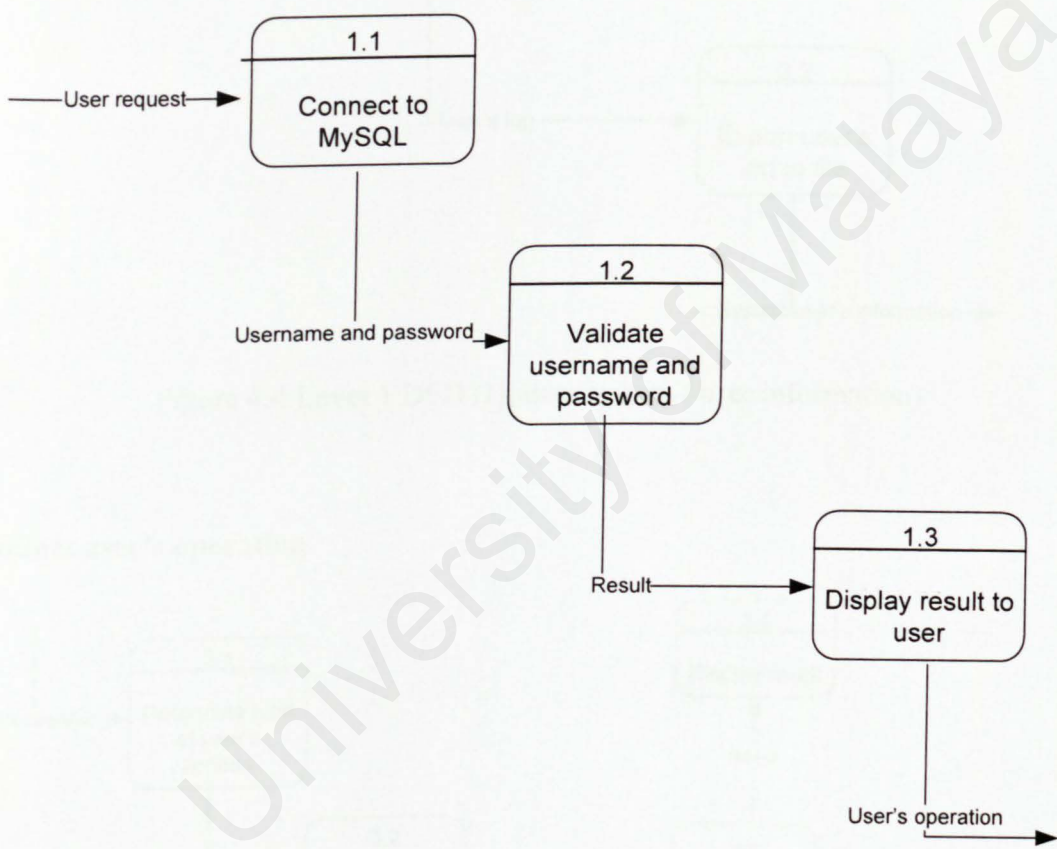


Figure 4.3 Level 1 DFD (Authenticate user)

Update system / user information

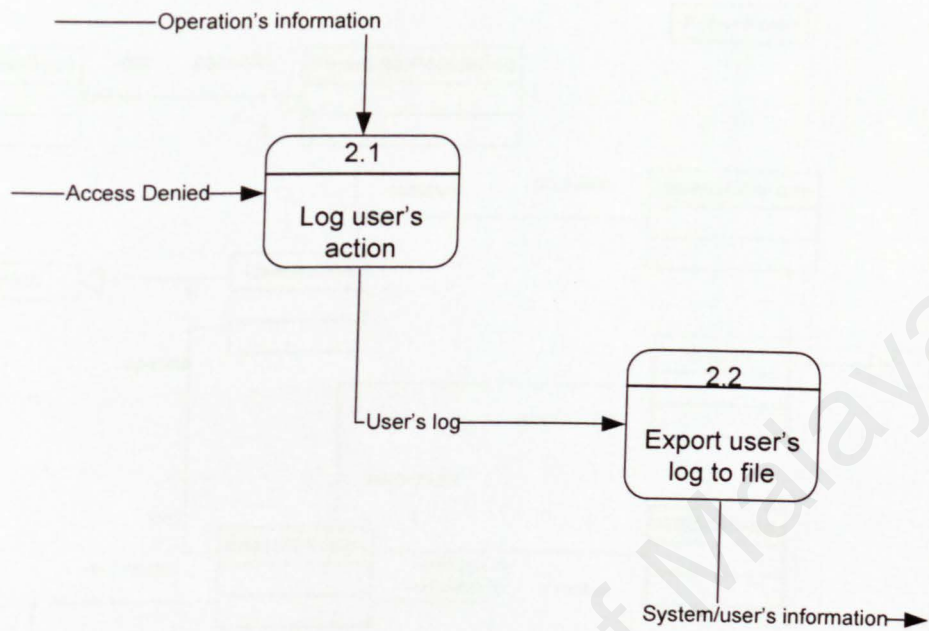


Figure 4.4 Level 1 DFD (Update system / user information)

Conduct user's operation

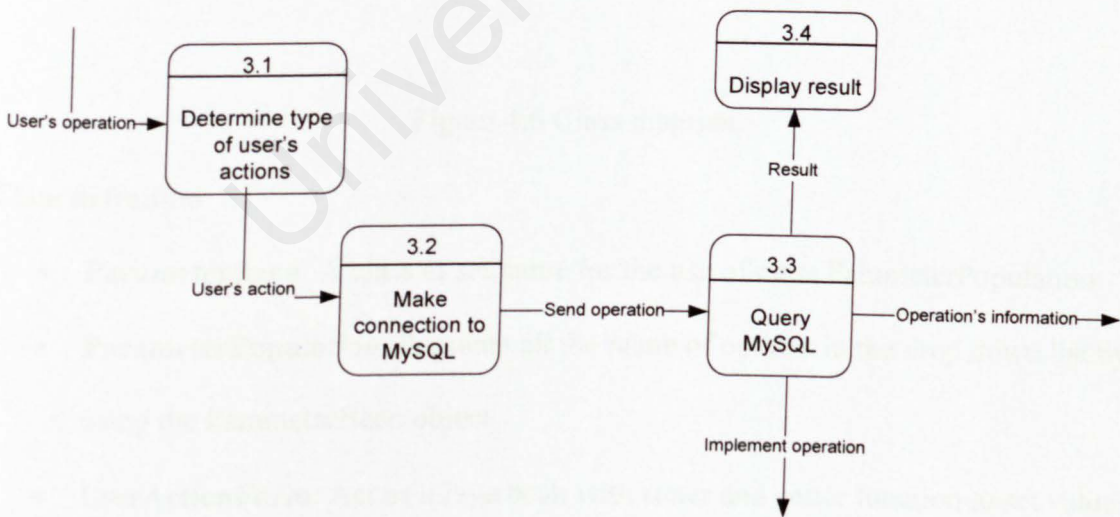


Figure 4.5 Level 1 DFD (Conduct user's operation)

4.3.4 Class diagram

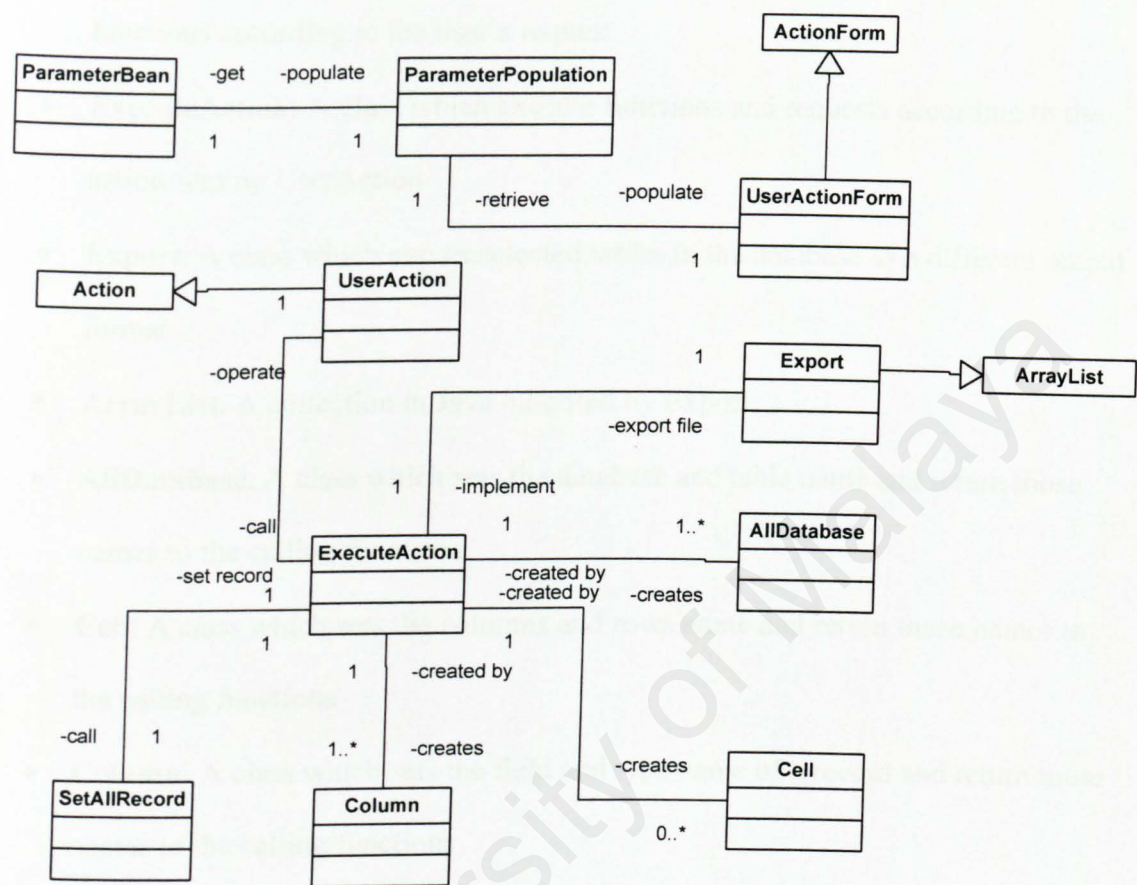


Figure 4.6 Class diagram

Class definition

- **ParameterBean**: A class to set name for the use of class ParameterPopulation
- **ParameterPopulation**: Populate all the name of options in the drop down list by using the ParameterBean object
- **UserActionForm**: Act as a Java bean with setter and getter function to set values entered by user from the form and return values to any calling function
- **ActionForm**: An abstract class in Struts inherited by the UserActionForm

- **Action:** A class in Struts inherited by the `UserAction`
- **UserAction:** A class which determines user's action and execute the appropriate functions according to the user's request
- **ExecuteAction:** A class which execute functions and requests according to the action sent by `UserAction`
- **Export:** A class which export selected tables in the database as a different output format
- **ArrayList:** A collection in Java inherited by `Export`
- **AllDatabase:** A class which sets the database and table name and return those names to the calling functions
- **Cell:** A class which sets the columns and rows name and return those names to the calling functions
- **Column:** A class which sets the field and type name of a record and return those names to the calling functions
- **SetAllRecord:** A class which sets records in a list and returns list to the calling functions

4.3.5 Sequence diagrams

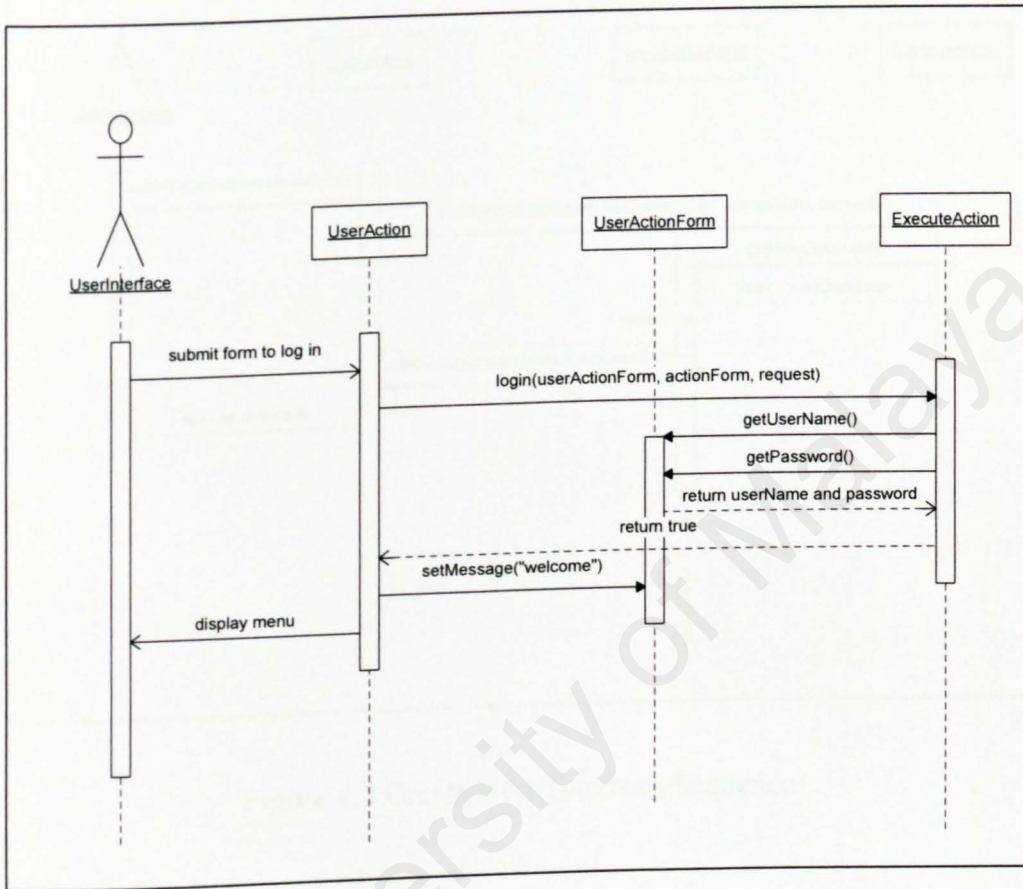


Figure 4.7 User Log In Sequence

When user submits his username and password in the login page, class ExecuteAction will be called and the function `login(userActionForm, actionForm, request)` will be triggered. This function will get the username and password entered by the user which was saved in the UserActionForm and try to connect to the MySQL. If the password is correct, it will return a true value, else false will be returned

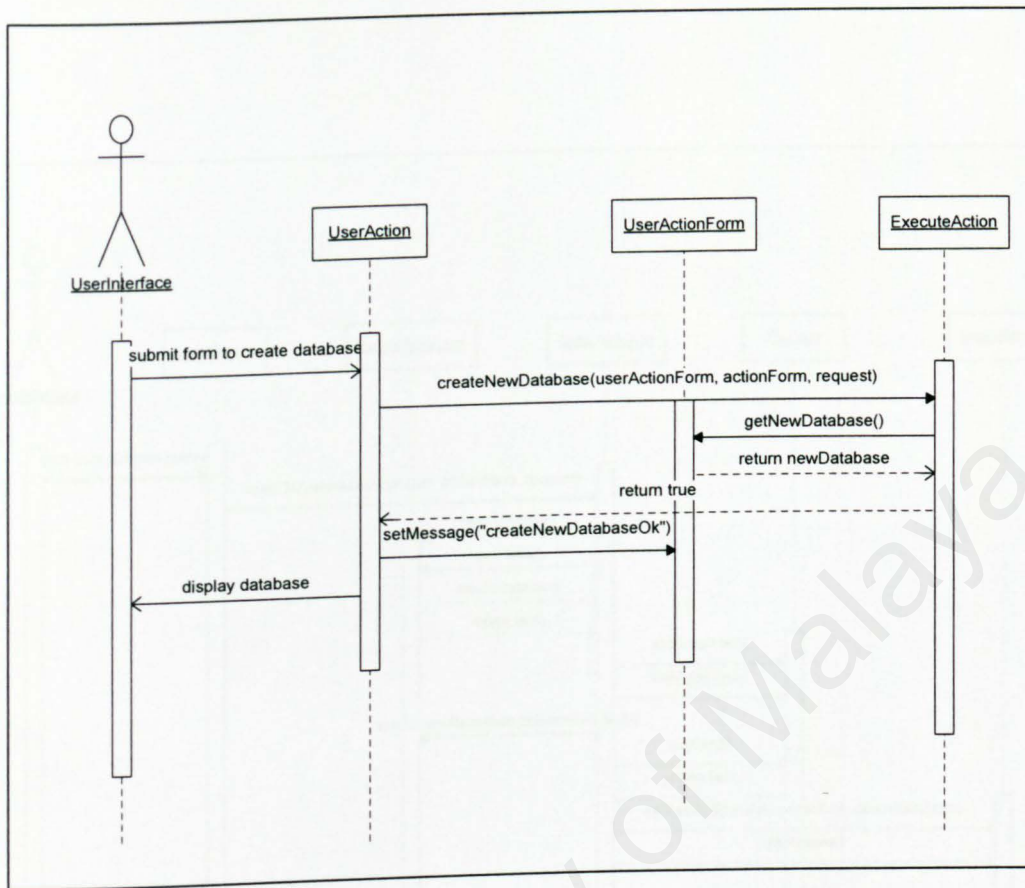


Figure 4.8 Create New Database Sequence

When user submits form to create a new database, function `createNewDatabase(userActionForm, actionForm, request)` of `ExecuteAction` will be called. The database name entered by the user will be retrieved from `UserActionForm` and query to create the new database with the name submitted by the user will be executed. If all the operations is a success, it will return true, else false will be returned.

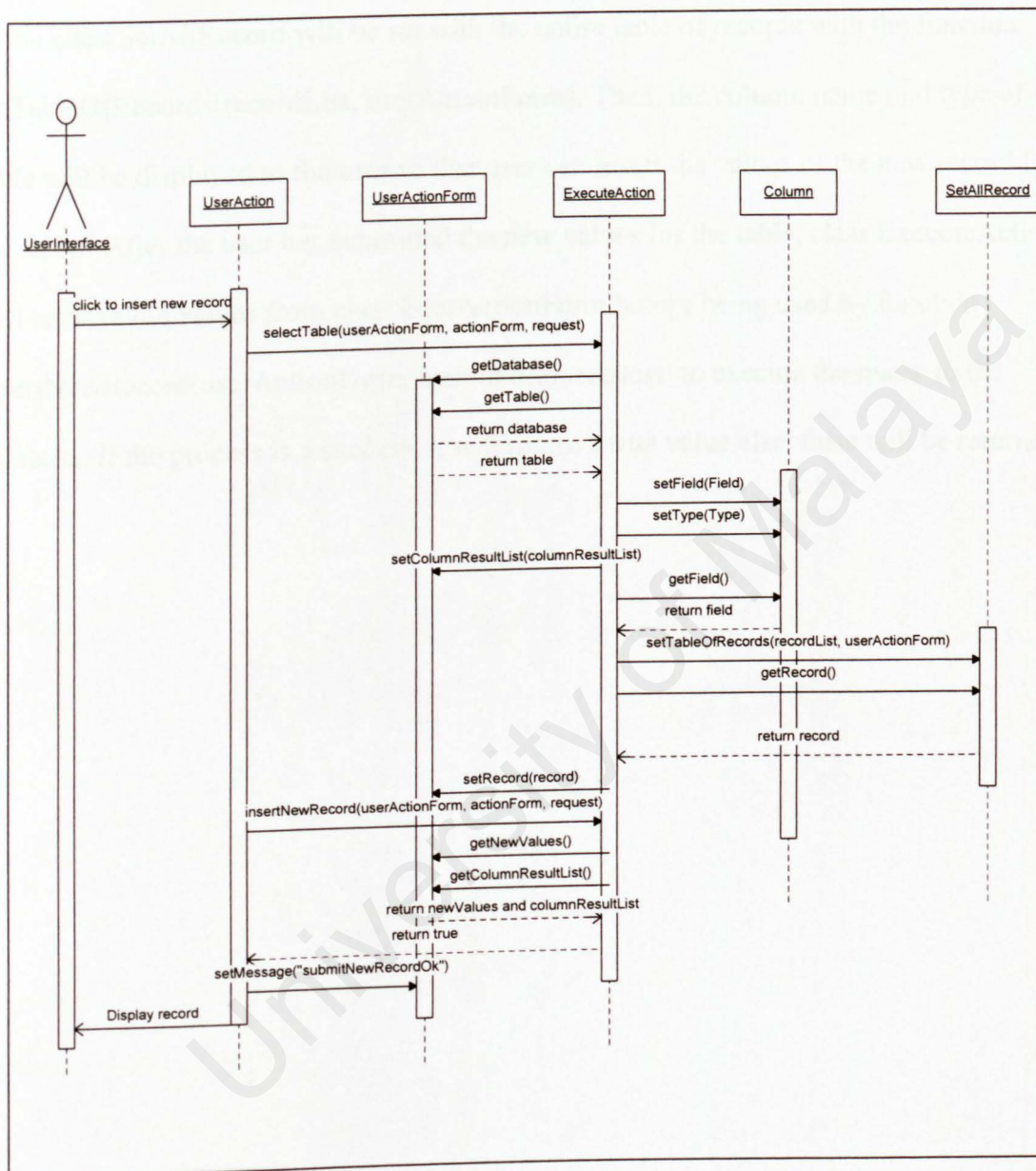


Figure 4.9 Insert New Record Sequence

When user clicks to create a new record, class ExecuteAction will retrieve the database and table name from the UserActionForm. Then, function selectTable(userActionForm,

actionForm, request) will query the database to gather all the records of the selected database and table. Class Column will be set for the appropriate field and type name while class SetAllRecord will be set with the entire table of records with the function setTableOfRecords(recordList, userActionForm). Then, the column name and type of the table will be displayed to the user so that user can insert the values of the new record for the table. After the user has submitted the new values for the table, class ExecuteAction will retrieve the values from class UserActionForm before being used by function insertNewRecord(userActionForm, actionForm, request) to execute the query to the database. If the process is a success, it will return a true value else, false will be returned.

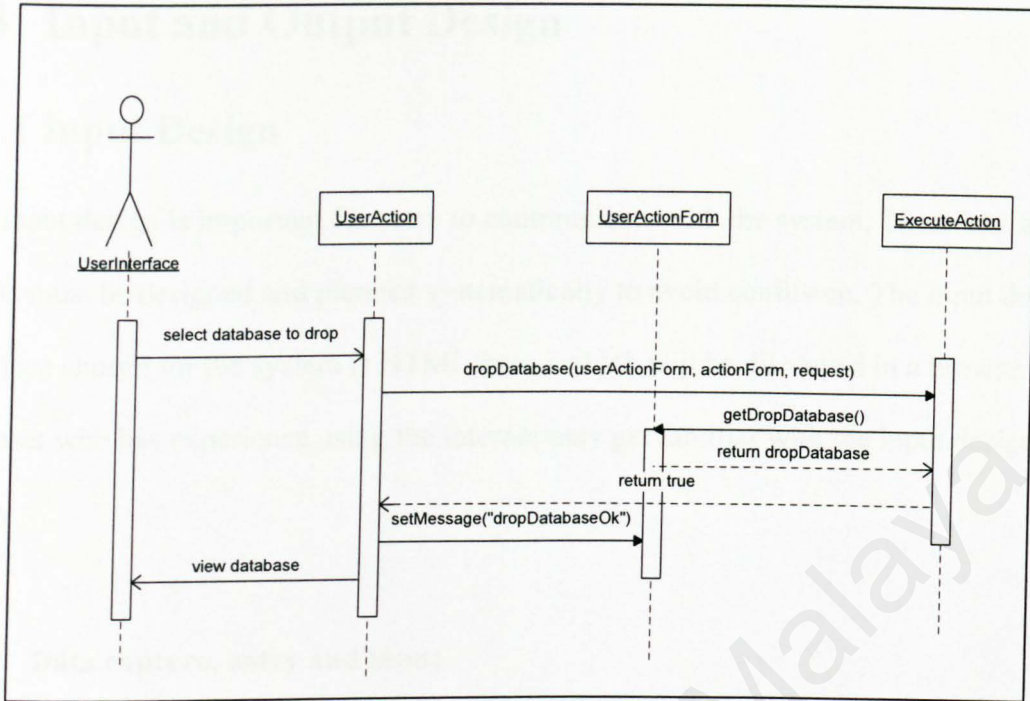


Figure 4.10 Drop Database Sequence

User will select the database to be dropped and submit it to the UserAction class. Then, the function dropDatabase(userActionForm, actionForm, request) from ExecuteAction will be called and it will retrieve the user selected database value from UserActionForm. The drop database query will be executed in the dropDatabase function and if the process is a success, it will return a true value, else false will be returned.

4.4 Input and Output Design

4.4.1 Input Design

The input design is important for users to communicate with the system. Therefore, all inputs must be designed and planned systematically to avoid confusion. The input design interface chosen for the system is HTML-based which will be displayed in a browser. So, any user who has experience using the internet may get familiar with the input design easily

- **Data capture, entry and input**

User will be providing inputs in text form which is entered in text fields or text area. Drop down list will be used to let users choose specific choices while checkboxes will be used to gather user's multiple choices. Buttons and links will be used as a source of navigating and submitting data throughout the system

- **Input validation**

Inputs which are entered by user will be checked to ensure all mandatory fields are entered. Then, inputs will be validated to check whether it fulfills the correct data type before being used by the system to perform the user's request. If the data entered does not fulfill any of these, users will be prompt to enter the data again by providing hints on what the error is

4.4.2 Output Design

The output design is meant to display and send the appropriate results to users depending on the inputs provided. All outputs can be divided into three forms:

- Display results in browser and users may see and work on it interactively
- Generate database in separated text file format and user may retrieve it for personal use
- Compress database in a separated zip file format and user may retrieve it for personal use.

4.5 Interface Design

The main idea of user interface design is simplicity and easy to view and use. Therefore, MySQLAdminPro interface design will meet this requirement by using general and straightforward approach with minimal navigation between forms and results.

- MySQLAdminPro is optimized to be displayed on Internet Explorer because it is a Windows platform system and Internet Explorer is the default browser for Microsoft Windows
- For optimal performance, graphical usage is minimal in interface design to shorten loading time
- Font style, format, color and themes of the design is standardized so that users may get used to the interface

- Terms and languages used are similar to the MySQL relational database management system so that users may understand the interface well and help them to master the system easily
- The interface is also very responsive with user's action and interactions with the system by acknowledging the user when a request has completed or in case of a failure during some illegal actions

Following are a few printscreens of the MySQLAdminPro interfaces:

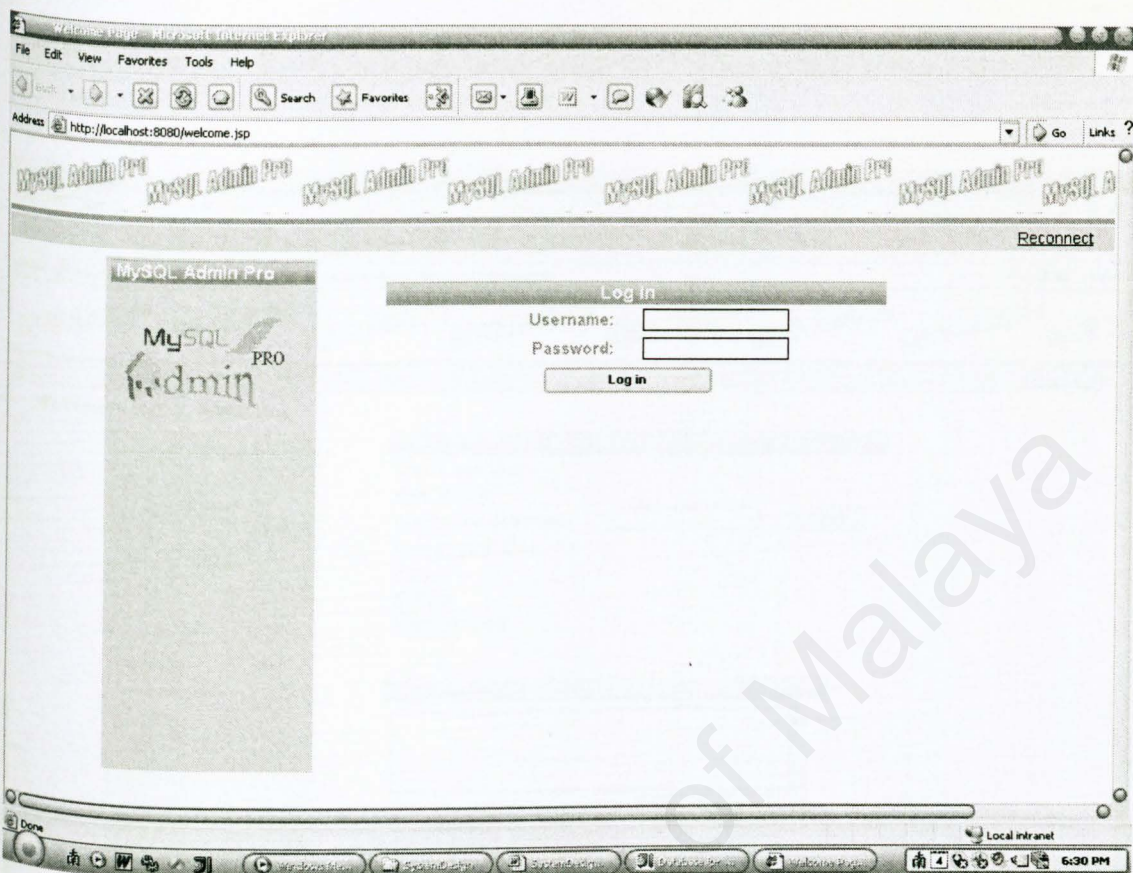


Figure 4.12 Log in menu

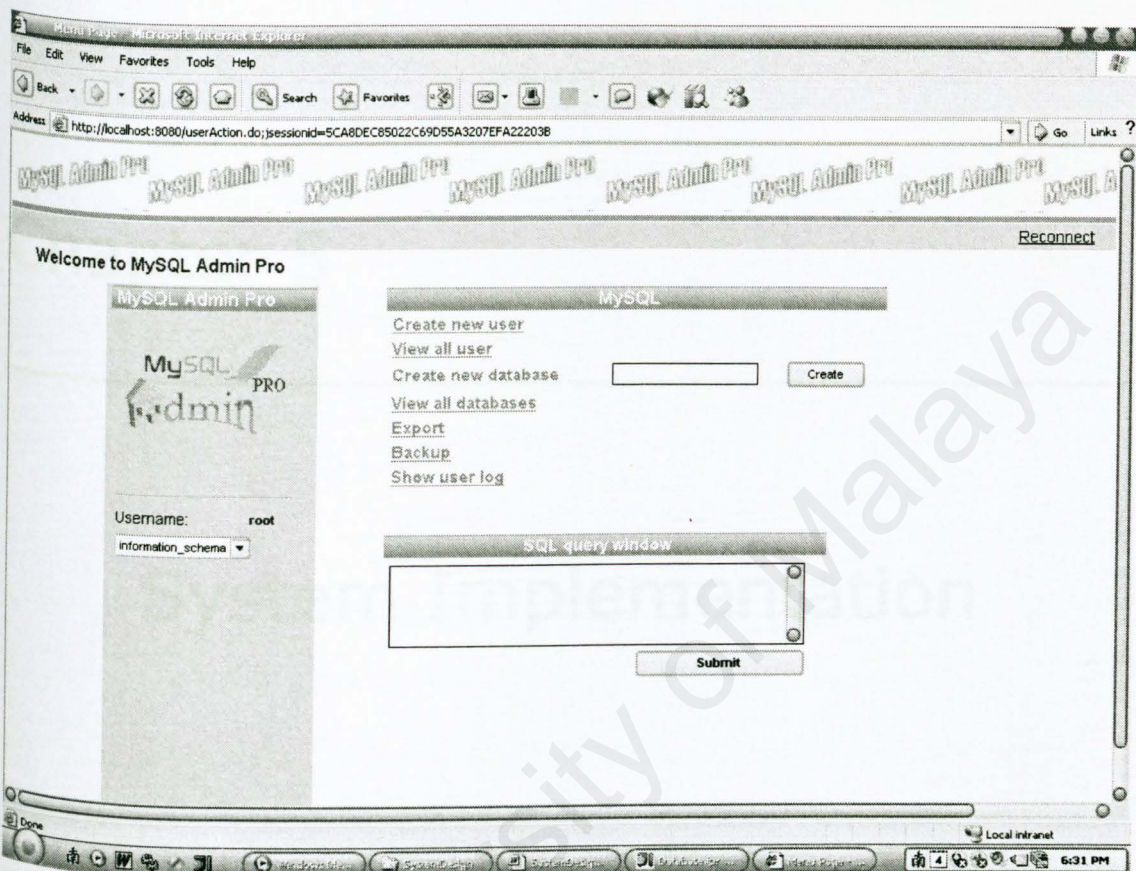


Figure 4.13 Main menu

Chapter 5

System Implementation

Chapter 5: System Implementation

5.1 Overall System Development Approach

MySQLAdminPro was developed using the object oriented approach where the fundamental elements of the system like the database table's column and cell were described in an object form. These objects were created consequently by evaluating all the functions that has to be delivered by the system.

All functions will then be scheduled according to its priority, time and effort needed. For example, functions with higher priority like adding a database has to be completed first before the drop database function is developed. Other functions like user's history logging will be developed after all other functions has completed so that all functions were involved in the logging process.

These objects, classes and the associated methods will then be used in the system design by drafting the Unified Modeling Language (UML) and Data Flow Diagram (DFD). The implementation (coding) began and progressed according to these UML and DFD design one by one. In another word, the development of the system proceeded by adding methods to the classes which were defined earlier to develop the required function one by one according to its priority.

5.2 Style and Design Approach

Since the presentation-tier of the system is on a browser, HTML was used as the formatting language. Thus, the style and design is described using the Cascading Style Sheets (CSS). The design was planned to be simple and less sophisticated, so bright fonts, white background and straightforward outline is chosen to deliver the user interface.

Among a few things which was covered in the CSS is the outline of the page, the type of fonts used, the color of the text, the appearance of buttons and links, the color of the background and the images which need to be included.

The same style and design were used in every page of the display to avoid confusion and increase the usability of the system. This can be done by creating an independent CSS file so that every HTML page can adopt the design from it by including the specified CSS file in the header of the HTML file.

5.3 Implementation with JavaScript

Every HTML page has its own JavaScript file. This JavaScript file is used to assign value to the property of the Java object which is used in the coding. For example, when an user click a button on the HTML page, the value of the button will be sent to the Java object using JavaScript and by determining this value, we will know that the user has actually clicked that particular button. Other than value parsing, JavaScript served another usage in the system. It is used to submit the entire HTML form whenever the user has finished his/her operation.

5.4 Implementation with IntelliJ IDEA Integrated Development Tool (IDE)

IntelliJ IDEA is an intelligent tool in Java development where it provides a lot of development utilities like (Java 2 Enterprise Edition) J2EE support, ANT, JUnit and version controls integration.

IntelliJ IDEA was chosen as the development tool for the system where it's used to perform coding with Java, as the code behind the business logic. Java Server Pages (JSP) which acted as the presentation-tier logic was coded with IntelliJ IDEA too. Besides, the IDE also supports JavaScript and Cascading Style Sheet (CSS) which was used to do client side processing and style-formatting.

5.4.1 Program commenting

The purpose of code commenting is to aid the code readability because more time is spend in reading the code than writing it. In the process of coding MySQLAdminPro business logic using Java, description of what the code can perform will be explained using code commenting which consist of “//” and “/* */”. An example of code commenting in MySQLAdminPro business logic is shown below:

```
/**
 * Create a new database
 *
 * @param userActionForm    Account Action Form
 * @param actionForm        Action form.
 * @param request            Http Servlet Request
 * @return boolean
 */

public boolean createNewDatabase(UserActionForm userActionForm
                                , ActionForm actionForm, HttpServletRequest request)throws
Exception
{
    try{
        Log.writeln("\nClass: ExecuteAction \nMethod: createNewDatabase\n");

        Class.forName("com.mysql.jdbc.Driver").newInstance();
        Connection con=getConnection(userActionForm, actionForm, request);

        String newDatabase = userActionForm.getNewDatabase();
        String statement= "create database '"+newDatabase+"'";
        Statement s=con.createStatement();
        s.executeUpdate(statement);
        return true;
    }
    catch(SQLException e){
        Log.writeln("\nClass: ExecuteAction \nStatus: createNewDatabase
(SQLException)\n");
        return false;
    }
}
```

Figure 5.1 Program Commenting

5.4.2 Exception handling

The purpose of exception handling is to catch the program's error or condition when something out of expectation has occurred. This is important to track what has gone wrong with the system and why does it behaves strangely during implementation. This situation will be logged to a file to be reviewed later and user will be notified that something out of expectation has occurred and will be guided by easy to understand messages as a response to tell them what to do next. An example of exception handling is shown in Figure 5.1 where the "catch" syntax will catch all the SQL generated exception

5.4.3 Import Libraries

Since Java is an object oriented programming language, there are a lot of libraries integrated in the software development kit which can be included to reduce the development time and effort. This is a good object reuse practice and an example of using this approach is shown below by importing the library file in the beginning of the code:

```
import org.apache.struts.action.ActionForm;
import javax.servlet.http.HttpServletRequest;
import java.io.*;
import java.sql.*;
import java.util.List;
import java.util.ArrayList;
import java.util.zip.ZipOutputStream;
import java.util.zip.ZipEntry;
import java.util.zip.Deflater;

public class ExecuteAction{
    .....
}
```

Figure 5.2 Import Libraries

5.4.4 Apache Struts Framework implementation

Apache Struts is a framework for deploying the servlet and JSP application which uses the (Model-View-Control) MVC architecture. This framework will separate the presentation-tier from the business logic tier so that all the presentation code (HTML code) will not be mixed together with the business logic code (Java code) so that each tier is clean without overlapping and development can be done easily and independently. Other than this, the Struts framework also provides a collection of utilities to handle many common tasks in web application development. These utilities can be made available in the development process by including the Struts package file in IntelliJ and import the specific library during coding by using the “import” syntax as shown in Figure 5.2

5.5 Implementing the Database

5.5.1 Setting up the Database

The MySQLAdminPro for sure, uses the MySQL database because its main purpose is to help and ease users in manipulating the database. Setting up the database needs nothing more than installing the MySQL database in the Windows environment and when this is done, MySQLAdminPro is ready to perform its operation on it

5.5.2 Database Connection

Database connection is performed using the help of the Java Database Connectivity (JDBC) library package. With JDBC, a connection can be set up in the business logic code to perform the requested manipulation with MySQL. In the system, one method is coded to handle the job of connecting to MySQL as shown below:

```
/**
 * Make connection with username and password
 *
 * @param userActionForm    Account Action Form
 * @param actionForm        Action form.
 * @param request           Http Servlet Request
 * @return Connection
 */
public Connection getConnection(UserActionForm userActionForm
                                , ActionForm actionForm, HttpServletRequest request)
throws Exception{
    Connection nullConnection = null;
    try{
        Log.writeln("\nClass: ExecuteAction \nMethod: getConnection\n");

        Class.forName("com.mysql.jdbc.Driver").newInstance();
        String userName = userActionForm.getUserName();
        String password = userActionForm.getPassword();
        Connection con=DriverManager.getConnection("jdbc:mysql:///","userName ,
        password");
        return con;
    }
    catch(Exception e){
        Log.writeln("\nClass: ExecuteAction \nStatus: getConnection (Exception)\n");
        return nullConnection;
    }
}
```

Figure 5.3 Database Connection

5.5.3 Querying the Database

SQL language is used in querying the database to perform the requested operations. The SQL statement which was created initially will be executed using the 'Statement' object which is stored in java.sql library. An example of querying the database to create a new database is shown in Figure 5.1

5.6 System Modules and Functionalities

5.6.1 Access authentication

User of the MySQLAdminPro system must log in with their MySQL username and password. Username and password will be granted by the MySQL admin by creating a new user in the mysql database. Other than this, the admin can choose the suitable privileges to be granted to the newly created user. These privileges are divided into three categories which are 'data', 'structure' and 'administration' (Table 5.1). In another way round, the admin may delete users from the mysql database too. All available users of the MySQL database can be viewed in the 'View all users' page too.

Table 5.1 User Privileges

Privileges		
Data	Structure	Administration
SELECT	CREATE	GRANT
INSERT	ALTER	SUPER
UPDATE	INDEX	PROCESS

DELETE	DROP	RELOAD
FILE	CREATE TEMPORARY FILES	SHUTDOWN
		SHOW DATABASES
		LOCK TABLES
		REFERENCES
		EXECUTE
		REPLICATION CLIENT
		REPLICATION SLAVE

5.6.2 Database and table manipulation

Create and delete database

Database can be both created and deleted by the user provided that the user has the right privileges which can be set by the admin. If the user submits a database which has already existed, an error message will be prompt. All databases can be deleted in the 'View all databases' page.

Create and delete table

Similarly, tables can be created and deleted by the user provided that the user has the right privileges which can be set by the admin. If the user submits a table which has already existed, an error message will be prompt. The user may also enters the column name, the column type, its length, its charset and also select the table's primary key when creating a new table.

Create, edit and delete record

Records can be created and deleted to the table provided that the user has the right privileges which can be set by the admin. These records in the table can be viewed page by page and each page is occupied by ten records.

Export database and table

Records in tables can be exported to the text or Microsoft Excel format using the export function.

Compress database and table

The whole database and its tables can be compressed to a zipped file format. This file contains the SQL script which can be used to populate back the databases and tables

5.6.3 User's history logging

All classes and methods which have been accessed by the user the moment he/she uses the system will be logged to a text file. The admin can check the file for user's past history or uses the file to track errors and exceptions encountered when using the system

5.7 Summary

The MySQLAdminPro was built using Microsoft Windows XP as its operating system along with Apache Tomcat as its web server container and Internet Explorer for its presentation-tier to provide implementation and testing. IntelliJ IDEA acts as both the programming tool and to integrate the whole system together.

Java Database Connectivity (JDBC) provides database access technology and MySQL is the targeted database which is needed by the system for manipulation and operation.

Java Server Pages (JSP) and the Apache Struts framework technology main purpose is to separate the presentation from business logic so that development of the system can be done in a neat and systematic way.

Chapter 6

System Testing

Chapter 6: System Testing

6.1 Objectives of Testing

Testing is one of the most vital phases in the software development lifecycle. Its purpose is for quality control where it checks or validates whether the system is performing as it is expected without any doubts or exceptions. It is best to perform testing for the system thoroughly after each module or unit has been completed to ensure that the errors or defects will not affect the next module or unit during development.

Errors or defects will be debugged to identify its cause so that it can be repaired or fixed. Some of the techniques of debugging are like performing traces in the code, identify the algorithm and function's correctness and identify the error states and exceptions. After these errors have been removed, it is important to retest the module to ensure that similar errors will not occur and other types of error will not exist and affect other modules after the modification.

6.2 Test Strategies

In the process of MySQLAdminPro development, four test strategies will be used to determine the level of success of the system. The five strategies are:

- Unit testing
- Data validation testing

- Integration testing
- System testing
- User acceptance testing

6.2.1 Unit Testing

Unit testing involves testing the modules of the system before they are integrated with other components of the system. All internal data, structures, logic and conditions will be tested with some input data and the output will be evaluated and compared against the expected result

In the development of MySQLAdminPro, tests are performed on each link, button, drop-down menu, checkbox and textbox to ensure that each of these elements is working correctly. Then, input test data will be entered and sent to the respective functions, and the output will be checked to make sure that each of these elements are correctly link to each of the functions to produce the expected result

For example, the database and table creation module will be tested by creating different types of tables with different column types and character set. The module will be tested too by purposely submitting some incorrect data to check whether the module will handle the incorrect data properly

6.2.2 Data Validation Testing

This testing is to validate whether the user is entering the correct input type. In the system, there are functions which need the correct input type and length for operation. Therefore, validation will be done before the input data is used and if these data is not correct in terms of type and length, user will be prompted to reinsert the data. For example, if a character or a string is entered in the text box which expects an integer value, user will be prompted to reinsert an integer value

6.2.3 Integration Testing

Integration testing involves putting and combining the specific module with other modules and interfaces. Emphasis of this testing is on whether the specific module interacts as expected with other components

This testing also involves integration with the user interface to prove that elements on the user interface can trigger the specific function of the module correctly. It is also important to ensure that all modules and links of the pages are connected accordingly

6.2.4 System Testing

System testing analyzes the whole system and evaluates it as a whole. It involves the tester to perform all the functions which is stated in the requirements and evaluates the result of the test. It also involves testing on compatibility with other software or programs which is used by the system.

For MySQLAdminPro, this test is performed on the last phase of the development process. This is because all modules and integration between those modules need to be fully functional before the test can begin. The system is tested by activating all its functions and operations and ensures that the result is as expected.

Performance testing

Performance testing which is part of the system testing will evaluate the performance of the system on whether it conforms to the specifications and the non-functional requirements as below:

- **Usability**

Usability of MySQLAdminPro has been taken into consideration when designing the user interface. This is because MySQLAdminPro's main consideration is to assist user in manipulating the MySQL database and thus, an easy to use user interface is important. So, the design of the interface has been made simple with easy to understand commands and functions and being tested with end users who have limited knowledge of the MySQL database to make sure that it is simple and comfortable to use.

- **Reliability**

Reliability is the probability and capability of the system to be used for a period of time without any failure. Testing of MySQLAdminPro is conducted by submitting various input parameters and confirms that the system performs as expected.

- **Security**

Only users which have been added by the admin are allowed to use the system. Sensitive information like the user's password is encrypted and testing is done by using various users with different privileges to trigger all the functions provided. It is also important to check that only users with the right privileges granted can execute the specified functions.

6.2.5 User Acceptance Testing

The purpose of user acceptance testing is to ensure that the system is able to function fully in the intended environment and to make sure end users knows how to operate the system comfortably.

The user acceptance testing of MySQLAdminPro is done by a few friends of the author to get feedbacks and comments regarding the system for future enhancement. Generally, all end users are satisfied with the functions and operations of MySQLAdminPro

6.3 Test Plan

6.3.1 Unit Testing

Test Plan for MySQLAdminPro			
Name of tester: Wong Teng Foong			
Test strategy: Unit testing			
Testing module: MySQLAdminPro (Login)			
Place: Tester's house			
No	Condition tested	Expected result	Actual result
1	Enter valid username and password	Login successfully	Same as expected
2	Enter invalid username and password	Display error message	Same as expected

Test Plan for MySQLAdminPro			
Name of tester: Wong Teng Foong			
Test strategy: Unit testing			
Testing module: MySQLAdminPro (Create new database)			
Place: Tester's house			
No	Condition tested	Expected result	Actual result
1	Enter a valid database name	Database created successfully	Same as expected
2	Entered an existed database name	Display error message	Same as expected
3	Did not enter anything and 'Create' button is clicked	Display error message	Same as expected

Test Plan for MySQLAdminPro			
Name of tester: Wong Teng Foong			
Test strategy: Unit testing			
Testing module: MySQLAdminPro (Create a new table in existing database)			
Place: Tester's house			
No	Condition tested	Expected result	Actual result
1	Enter a valid table name and number of columns in the table	Prompt to insert information of table	Same as expected
2	Enter an existed table name or invalid column number	Display error message	Same as expected
3	Did not enter anything and 'Create new table' button is clicked	Display error message	Same as expected
4	Enter all the table's information correctly and 'Submit' button is clicked	Table created successfully	Same as expected
5	Did not enter the table field name or entered a duplicated field name	Display error message	Same as expected
6	Enter the wrong length for data type	Display error	Same as expected
7	Choose the wrong character set for data type	Display error	Same as expected
8	Did not select a primary key	Display error	Same as expected

Test Plan for MySQLAdminPro			
Name of tester: Wong Teng Foong			
Test strategy: Unit testing			
Testing module: MySQLAdminPro (Backup and compress database)			
Place: Tester's house			
No	Condition tested	Expected result	Actual result

1	Select a database to backup and click 'Compress'	Database successfully backup and compressed	Same as expected
2	Did not select a database or table and click 'Compress'	Display error message	Same as expected

Test Plan for MySQLAdminPro

Name of tester: Wong Teng Foong			
Test strategy: Unit testing			
Testing module: MySQLAdminPro (Export database table to another file format)			
Place: Tester's house			
No	Condition tested	Expected result	Actual result
1	Select a database to export, select the format and click 'Export'	Database successfully exported to C:	Same as expected
2	Did not select a database or table and click 'Export'	Display error message	Same as expected
3	Did not select the file format	Display error message	Same as expected

6.3.2 Data Validation Testing

Test Plan for MySQLAdminPro			
Name of tester: Wong Teng Foong			
Test strategy: Data validation testing			
Testing module: MySQLAdminPro			
Place: Tester's house			
No	Condition tested	Expected result	Actual result
1	Enter invalid information	Display error message	Same as expected
2	Enter invalid data type of input	Display error message	Same as expected
3	Required field not entered	Display error message	Same as expected

6.3.3 Integration Testing

Test Plan for MySQLAdminPro			
Name of tester: Wong Teng Foong			
Test strategy: Integration testing			
Testing module: MySQLAdminPro			
Place: Tester's house			
No	Test case	Expected result	Actual result
1	Log in redirection	menu.jsp is loaded	Same as expected
2	Create new user redirection	createNewUserPage.jsp is loaded	Same as expected

3	View all user redirection	viewAllUserPage.jsp is loaded	Same as expected
4	View all databases redirection	viewAllDatabasePage.jsp is loaded	Same as expected
5	View all tables redirection	tablePage.jsp is loaded	Same as expected
6	Export database redirection	export.jsp is loaded	Same as expected
7	Backup database redirection	compressPage.jsp is loaded	Same as expected
8	Show user log redirection	MySQLAdminPro.txt log file is loaded	Same as expected
9	Create new table redirection	createTablePage.jsp is loaded	Same as expected
10	Insert new record redirection	insertRecordPage.jsp is loaded	Same as expected
11	Edit record redirection	editTableRecordPage is loaded	Same as expected

6.4 Summary

No	Test strategy	Testing module	Remarks
1	Unit testing	All modules	All functionalities of MySQLAdminPro were tested and the test results match the expected results
2	Data validation testing	All modules	All data will be checked before being submitted to be used to avoid functional failures
3	System testing	Entire system	All functions meet the user requirements and all non-functional requirements are fulfilled
4	Integration testing	Entire system	All page navigations must be done without any error and all modules should interact with each other correctly
5	User acceptance testing	Entire system	End users are satisfied with the system

Chapter 7:

System Evaluation and Conclusion

Chapter 7: System Evaluation and Conclusion

7.1 Introduction

The purpose of system evaluation is to ensure that all user requirements are correctly and accurately built and developed in the system. This is important because minor errors can still be corrected if detected before launching the whole system. Other than that, the strengths and limitations of the whole system can be reviewed so that users are aware of what they can do with the system and what future enhancements can be added.

7.2 System Strengths

7.2.1 Security and Privileges

MySQLAdminPro is password protected where users need to enter their username and password before they are allowed to use the system. All passwords are protected and hidden behind the hashed code to guarantee the security and privacy of each user.

The admin of MySQLAdminPro may grant specific privileges when he is creating a new user account. In another word, the admin can set the limitations of what the user can do with MySQLAdminPro in database manipulation.

7.2.2 Conveniences in Database, Table and Record

manipulation

Creating, editing and deleting databases, tables and records are just a few clicks away.

The simplicity of the user interface which consists of links and buttons deliver high conveniences in user experience in functions and tasks handling. Other than that, easy to understand messages will be prompted whenever a function is triggered so that users may know that his/her actions and requests have been performed

7.2.3 Complete Database Backup and Compression

Backup is very important in case of disaster like virus attack or system failures which might cause data corruption. So, MySQLAdminPro is equipped with a backup function where user may backup his/her database into a script file. This script file so powerful as it can be used to regenerate the whole database along with its tables and data when it is run. Furthermore, MySQLAdminPro is prepared with the compression function where it can compress the large script file to a smaller size file.

7.2.4 Database Table Export

Table of databases can be exported to an external Microsoft Excel file or text file by using the export function of MySQLAdminPro. This is a very important function if user wishes to export the records in the table to an external file format for other usage.

7.2.5 User History Logging

Every action and function which is triggered by the user whenever he/she uses the system will be logged to an external log file. This file is very important to keep track of what the user has done with the system and vital for troubleshooting if errors or failures of the system occurs. This is because the log file also contains all error and exception messages which will be generated by the function in case of error happens

7.3 System Limitations

7.3.1 SQL Query Execution

The SQL query window can only execute one query in one time only. In another word, the query window will not execute if the user submit multiple line of SQL statements in one time.

7.3.2 Limited Choice of File Type to Export

The MySQLAdminPro only provides 2 file types which are the Microsoft Excel and text file export. This means that user can only choose on either he/she wants to export his table to Excel or text file format

7.4 Future Enhancements

7.4.1 Multiple SQL Queries Execution

The SQL query window execution can be modified to execute more than one SQL statement in one time. One of the benefits of doing this is the ease of running or executing script which contains a lot of SQL queries in it.

7.4.2 Multiple Choice of File Type to Export

Other than Microsoft Excel and text file format, the MySQLAdminPro export module can be enhanced by including other file format too like PDF or HTML. This increases the variety of file format to be chosen by the user

7.5 Problems Encountered and Solutions

A few problems were encountered during the whole process of MySQLAdminPro development

7.5.1 Settings and Configuration Difficulties

Setting up the environment for the MySQLAdminPro development consumes a lot of time because there are a lot of integrations to be done. Setting up the IntelliJ IDEA Integrated Development Environment (IDE) to work with Apache Tomcat and Java Development Kit (JDK) is quite complicated and needs some research. Other than this,

packaging the whole MySQLAdminPro system into a single Web Archive (WAR) file is quite hard to do without some study on application packaging.

7.5.2 Improperly Defined Project Scope

There were times when the author was not certain about the scope and the extent of the system. However, after numerous discussions with the supervisor, all priorities and main concerns of the system is understood and managed to be delivered

7.5.3 Lack of Language Mastery

The author has less than a year of experience in web application development and it is a challenge to master the Java programming language in a short period of time. This has kept the author from adapting the best coding practices during the system development. However, through self study and guidance from friends, this has helped the author to acquire acceptable level of knowledge in coding and system development to deliver all the required functions

7.6 Knowledge and Experience Gained

A lot of knowledge and experiences were gained throughout the whole process of MySQLAdminPro development. Among them are:

7.6.1 Improvement in System Development Practices

Developing the MySQLAdminPro opens a lot of opportunities for the author to learn more on system development in real practices rather than studying all the theories behind it in class. In terms of system development, preparations like setting up a web application system and Unified Modelling Language (UML) and Data Flow Diagram (DFD) drawing can be done personally to improve the skills on those respective practices

7.6.2 Improvement in Programming Skill

Having the chance to involve directly in development and coding the whole system from scratch really improves the programming skill of the author. This is because the only way to develop knowledge in programming is to do it personally. Other than programming, the author also learnt to structure and plans the whole system development according to its priorities and sequences.

7.6.3 Project Planning

Other than software development skill, the author also learnt the importance of project planning. The author needs to plan the development of the whole system according to its time schedule and priorities so that the project can run systematically and conforming to the user requirements

7.7 Reviews on Goals

The summary of objectives and goals of MySQLAdminPro which were discussed in Chapter 1 are listed below:

- To control and manipulate the MySQL database more easily with user interfaces instead of command line
- To increase efficiency and effectiveness in database handling
- To provide access authentication for security purposes
- To provide database backup and compression

The system has successfully met all the objectives and goals listed. Other than that, MySQLAdminPro also fulfills other purposes such as table record exporting and user history and error logging.

7.8 Conclusion

The process of developing the MySQLAdminPro is very challenging since this is the first system which is fully handled by the author. Besides, the system also meets a lot of competition from other similar system like PhpMyAdmin in terms of functionality.

Anyway, MySQLAdminPro has successfully fulfilled all the requirements given and effectively launched to serve its needs

Skills and knowledge on software development have been greatly improved while building this system. Other than that, the author also has the chance to feel and experience the responsibility which needs to be carried in order to productively complete the whole process of software development.

References

- [1] Giacomo, M.D. (2005). *MySQL: Lessons Learned on a Digital Library*
- [2] Eisenberg, A., Melton, J. (2004). *SQL:2003 has been published*
- [3] Paracha, M.A., Mohammad, S.N., Macfarlane, P.W., Jenkins, J.M. (2003),
Implementation of Web Database for ECG
- [4] Seltzer, M.I. (2005), *Beyond Relational Databases*
- [5] MySQL official reference manual (2005)
- [6] phpMyAdmin 2.6.4-dev Documentation (2005)
- [7] Kline, K., Kline, D. (2001). *SQL in a Nutshell*. O'Reilly & Associates, Inc.
- [8] Hoffer, J.A., George, J.F., Valacich, J.S. (2005). *Modern Systems Analysis and Design*
- [9] DuBois, P. (2003). *MySQL*. Sams Publication
- [10] Suehring, S. (2002). *MySQL Bible*. Wiley Publishing, Inc.
- [11] Kotonya, G., Sommerville, I. (2002) *Requirements Engineering*, John Wiley & Sons

Appendix A

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY UNIVERSITY MALAYA

A survey on MySQL and its tools experiences. Only candidates with acceptable knowledge of MySQL database are selected to conduct this survey

This survey is conducted to gather public views on MySQL database tools and how such a tool would help them in manipulating the MySQL database.

1. Do you think that the Relational Database Management System (RDBMS) is an important technology and should be learnt?
☐ Yes
☐ No
2. Are you having difficulties in using the MySQL (RDBMS) command line interface?
☐ Yes
☐ No
3. Are you familiar with SQL queries and know how to use them well? Please rate your familiarity of the SQL queries from 1-3 (1 – Poor, 2 – Moderate, 3 - Expert)
☐ 1
☐ 2
☐ 3
4. Which of the following tools have you used before to overcome the MySQL command line interface?
☐ MySQL Query Browser
☐ MySQL Control Center
☐ phpMyAdmin
☐ Never use any tool before (Proceed to question 7)
☐ Others, _____
5. What are the advantages of the tool which you have chosen from question 3? (Can pick more than one)

- ☐ Easy and comfortable to use
- ☐ Easy to learn and master
- ☐ Rich of functions
- ☐ Others _____

6. What are the weaknesses of the tool which you have chosen from question 3? (Can pick more than one)

- ☐ Confusing
- ☐ Difficult to learn
- ☐ Poor functions
- ☐ Others _____

7. How would you rate the tool you are using on question 3?

- ☐ Very good
- ☐ Moderate
- ☐ Needs improvement

8. If there is a tool which can handle and manipulate the MySQL database easily, will you give it a try?

- ☐ Yes
- ☐ No

9. Please rate the importance of the features below for a MySQL database tool (1 - Not important, 2 - Moderate 3 - Very important)

- ☐ Easy to use
- ☐ Lots of functions
- ☐ Adding, deleting and editing records
- ☐ Database compression
- ☐ Browsing databases and tables
- ☐ Save user's past history
- ☐ Setting user privileges
- ☐ Fast response

10. Specify a function(s) that you think a database management tool should have:

User Manual

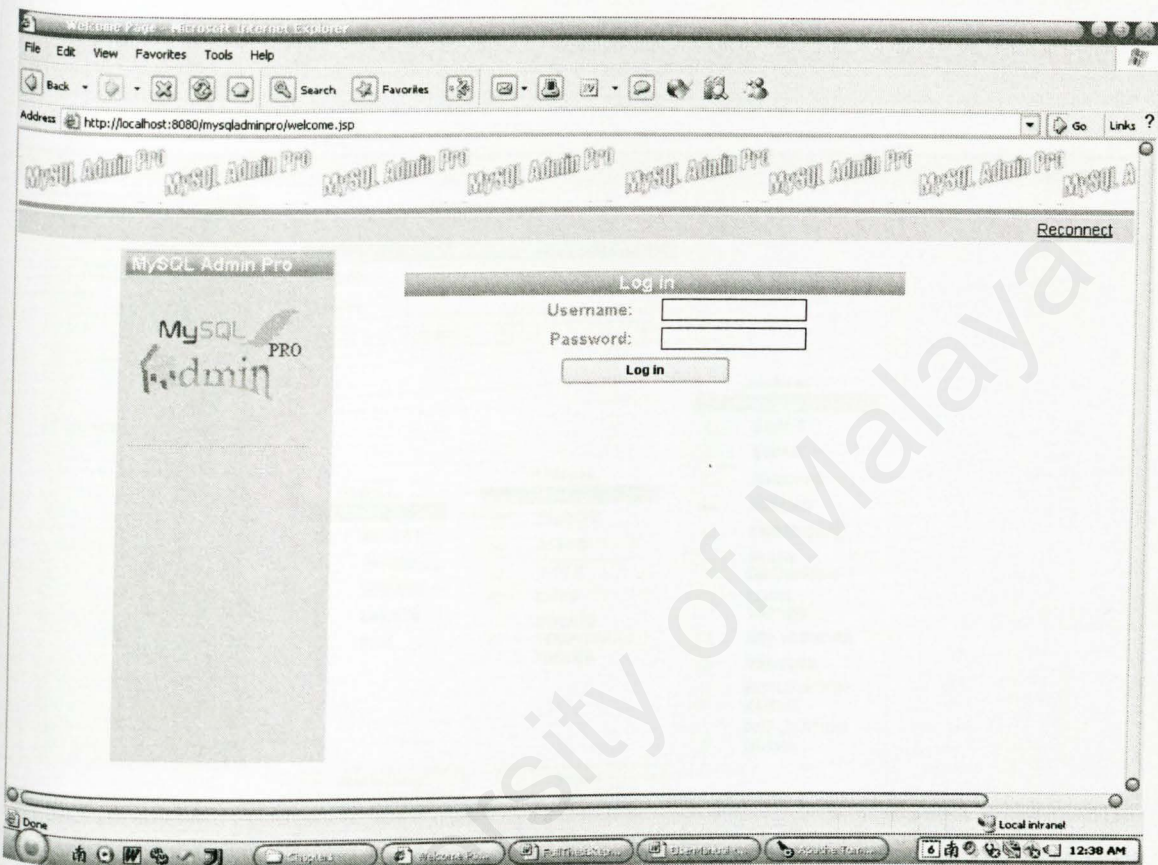
1.0 Prerequisite

- Make sure the system has the minimum hardware and software requirements specified. (3.3.3 Hardware requirements, 3.3.4 Software requirements)

2.0 Setting up MySQLAdminPro

- Install Apache Tomcat. The version which was used during development is Tomcat 5.0.30
- Install the MySQL database. Make sure the 'Include bin directory in Windows path' option is checked. The version which was used during development is MySQL Essential 5.0.15
- Copy the mysqladminpro.war file into the 'webapps' folder of Apache Tomcat.
- Start Apache Tomcat services
- Enter this URL on the browser address:
<http://localhost:8080/mysqladminpro/welcome.jsp> to start the system
- If page failed to start, try to restart the Apache Tomcat

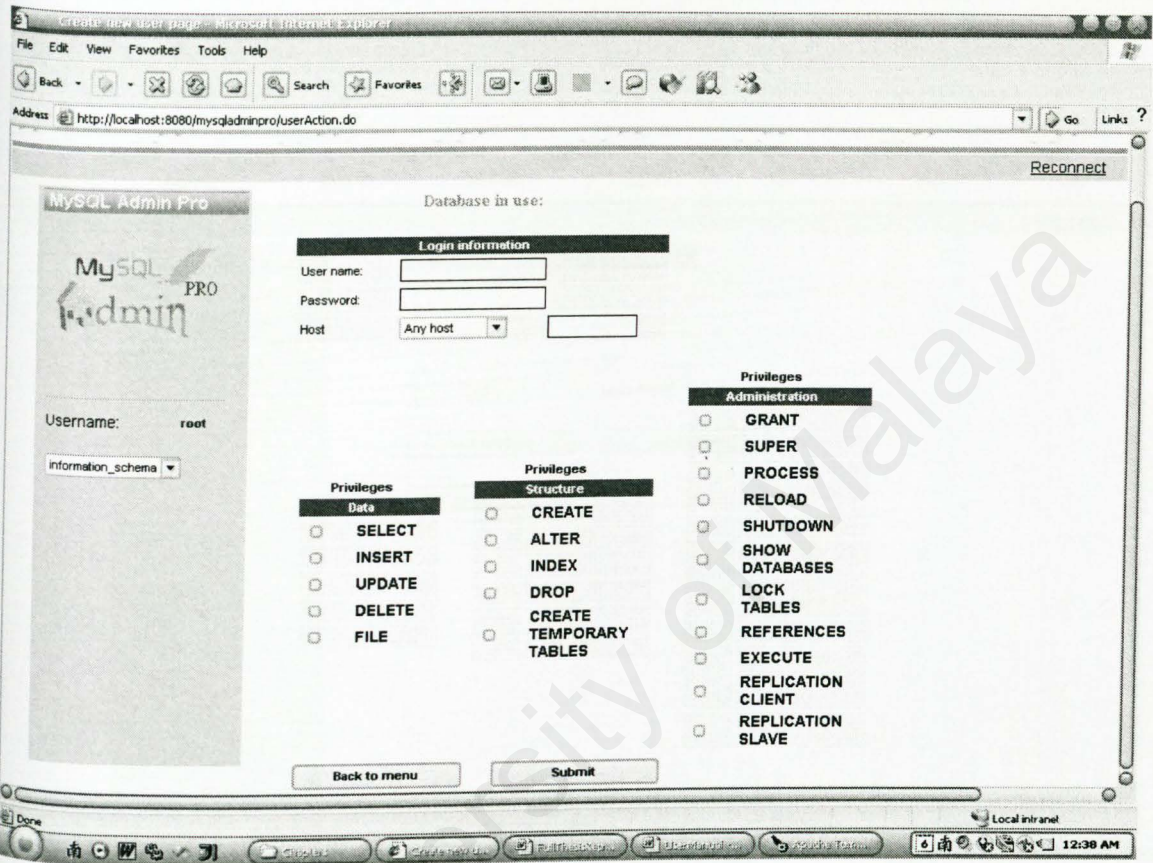
3.0 Login



- Enter the MySQL administrator username and password for admin login
- Enter the username and password created by the admin to login as other users.

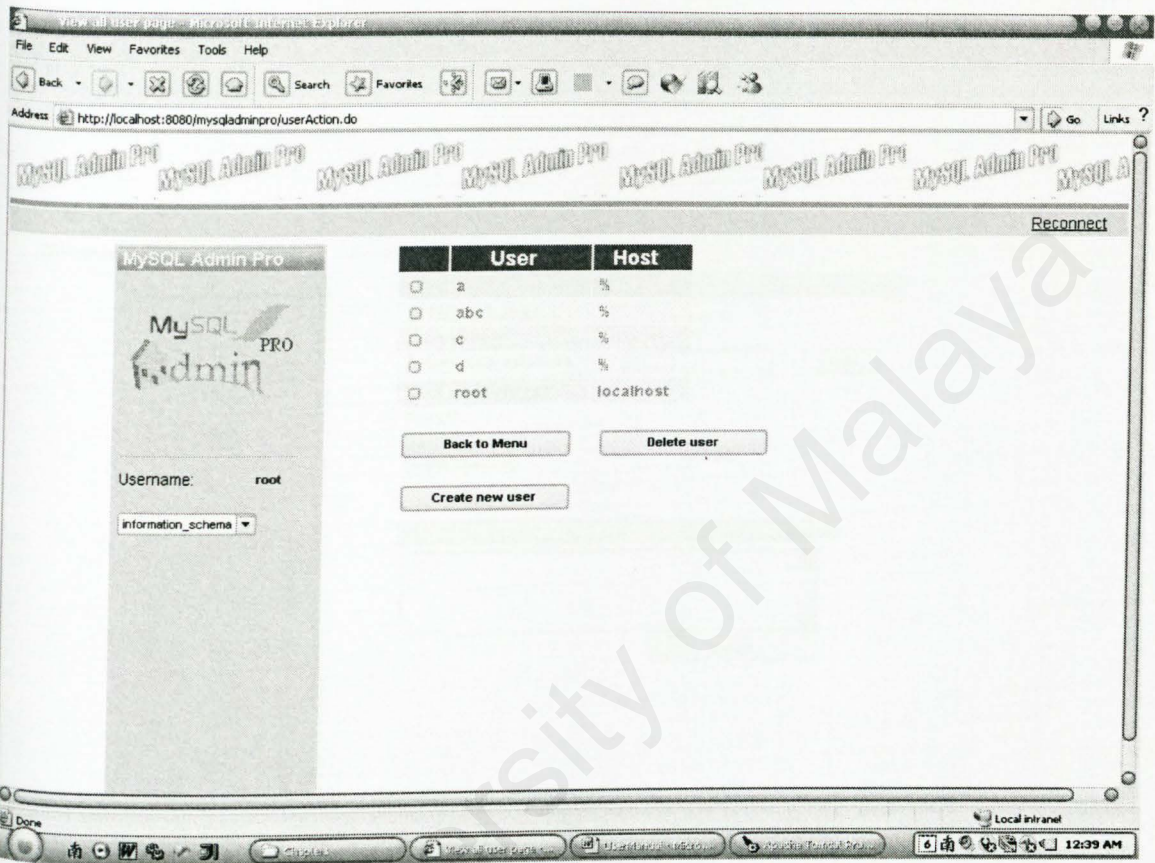
(Refer 3.0 on how to create new user)

4.0 Create new user



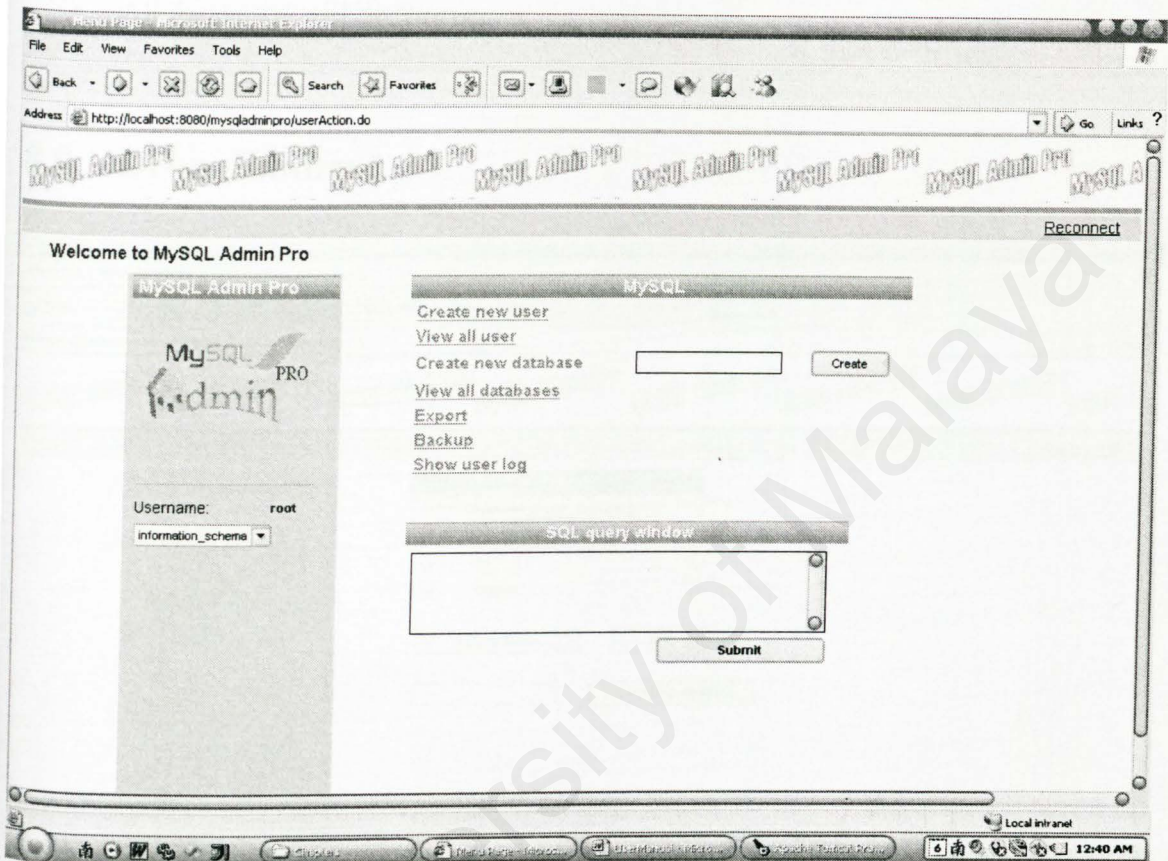
- Click the **[Create new user]** link on the menu page.
- Enter the username that you wish to create
- Enter the password for the user
- Select the host for the new user. For user specified host, enter the host name in the text box. For 'anyhost' and 'localhost', do not need to enter anything
- Select the privileges for the user
- Click **[Submit]** to create the new user

5.0 Delete user



- Click the **[View all user]** link on the menu page. This will show all available users of MySQL
- Select the users that you want to delete using the checkbox
- Click **[Delete user]** to remove the user
- Make sure the user has the privileges to delete user before doing it.

6.0 Create new database

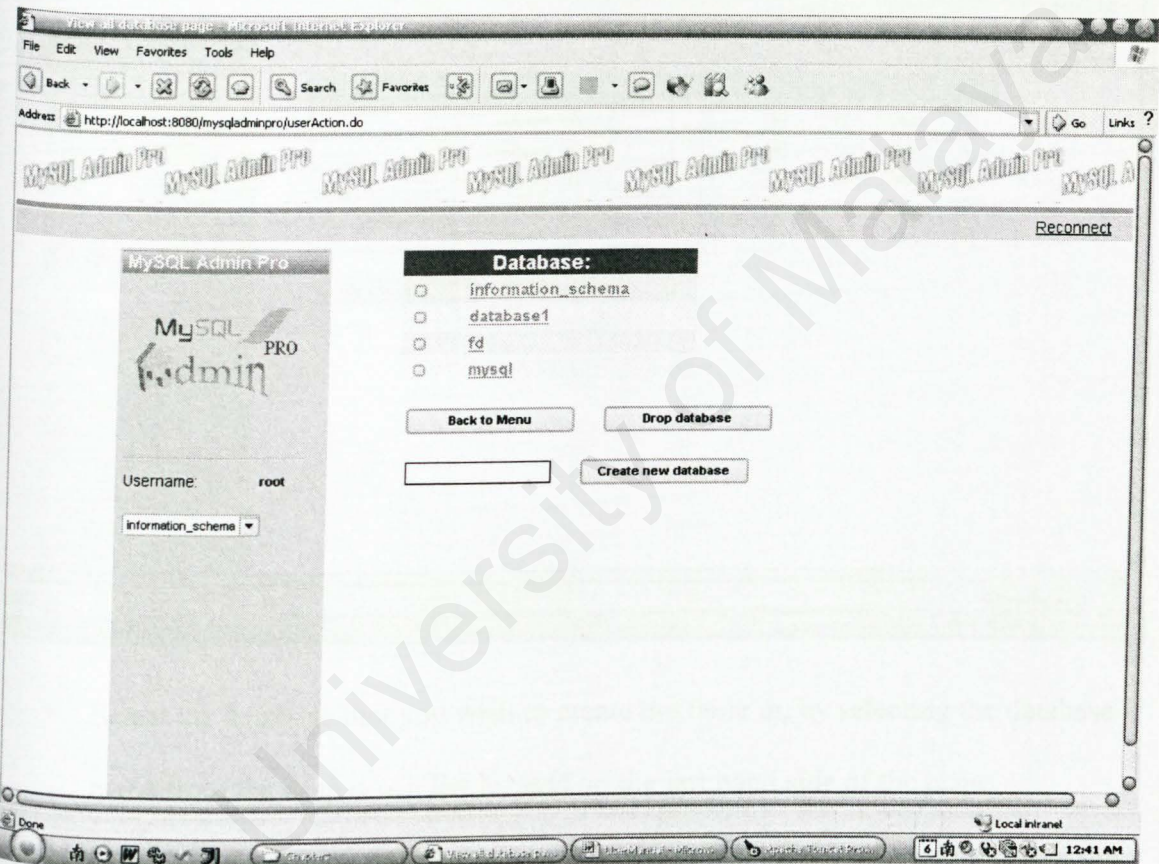


- Enter the database name that you wish to create in the **[Create new database]** text box on the menu page
- Click the **[Create]** button beside the textbox.
- Database can also be created in the 'View all databases page' (Refer to 6.0 on how to view all databases)

7.0 View all available databases

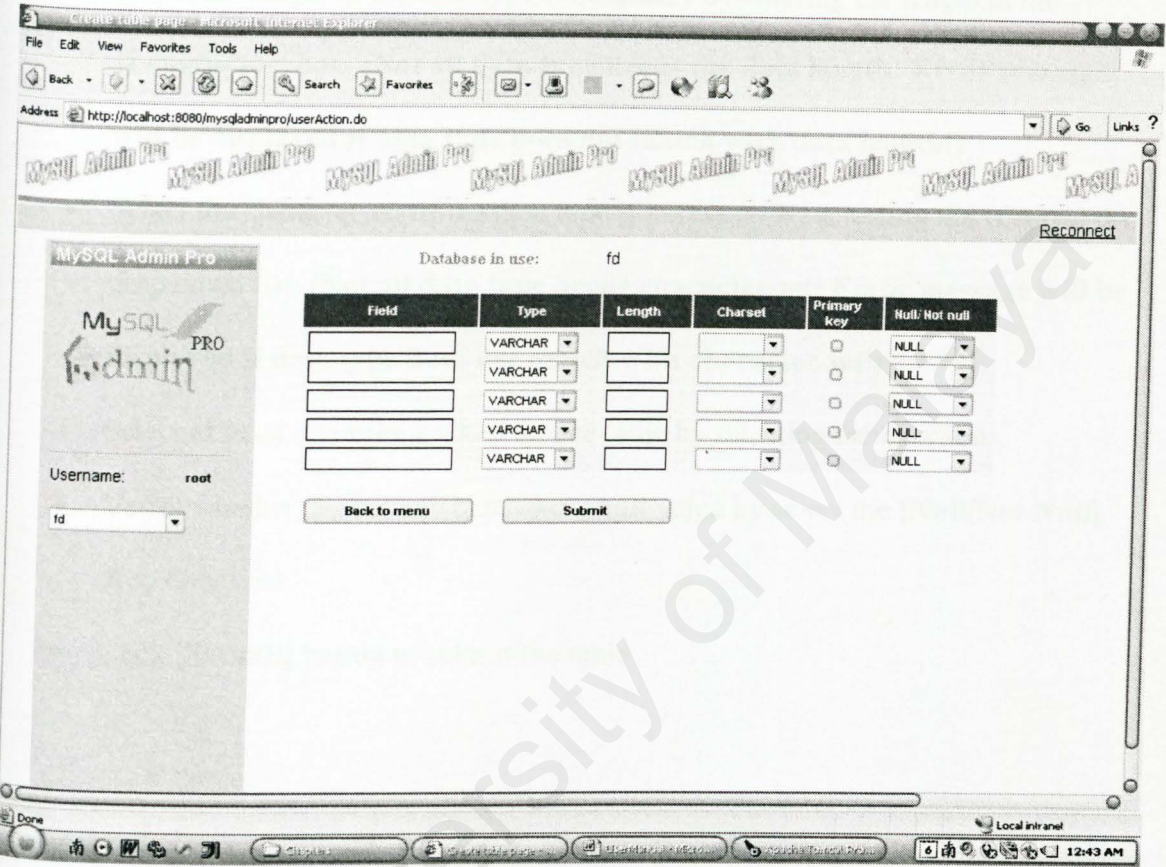
- Click the [View all databases] link on the menu page
- All databases will be shown

8.0 Drop database



- Click the [View all databases] link from the menu page
- Select the database that you wish to drop by selecting the checkbox
- Click the [Drop database] button

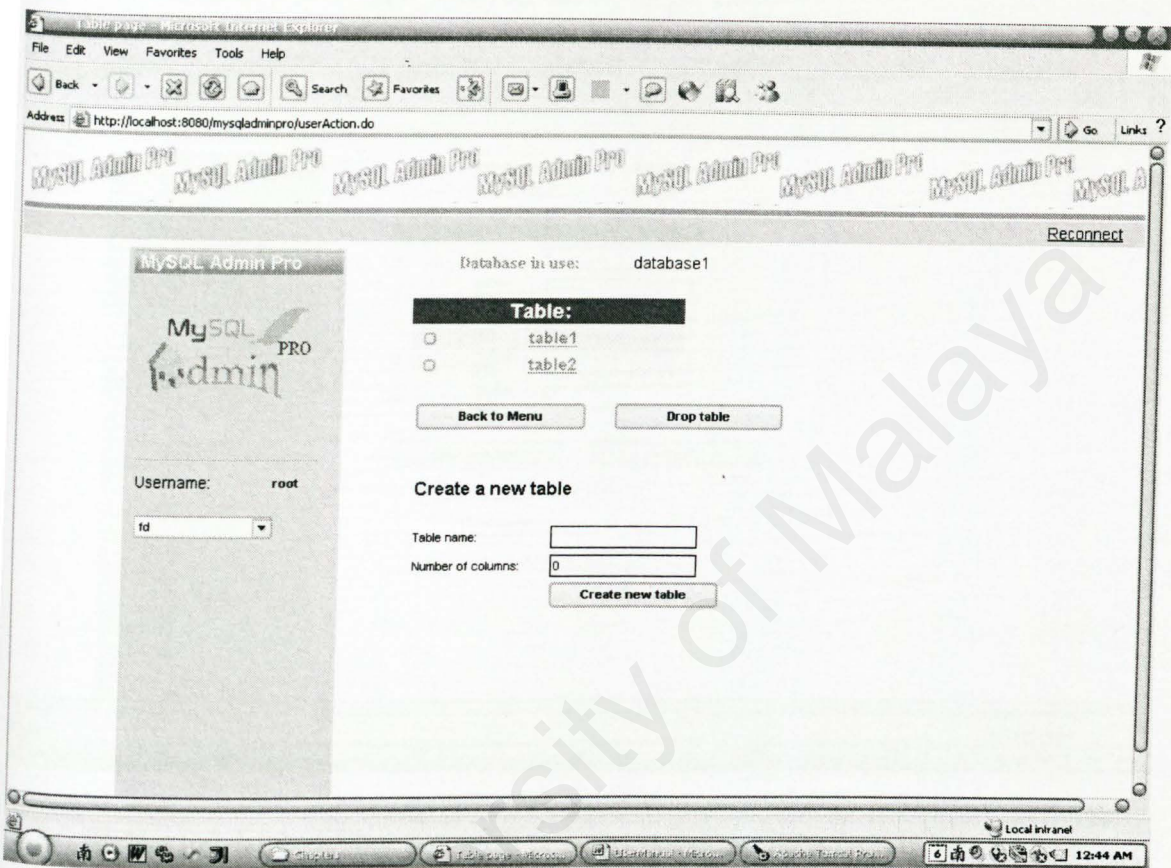
9.0 Create new table



- Select the database that you wish to create the table in, by selecting the database name from the drop down list located on the left hand side of the page.
- Database can also be selected by clicking the **[View all databases]** link from the menu page followed by the clicking the database you wish to create the table in.
- Enter the table name that you wish to create in the **[Table name]** text box
- Enter the number of columns that you wish the table to have in the **[Number of columns]** text box
- Click the **[Create new table]** button to submit the table name and its columns

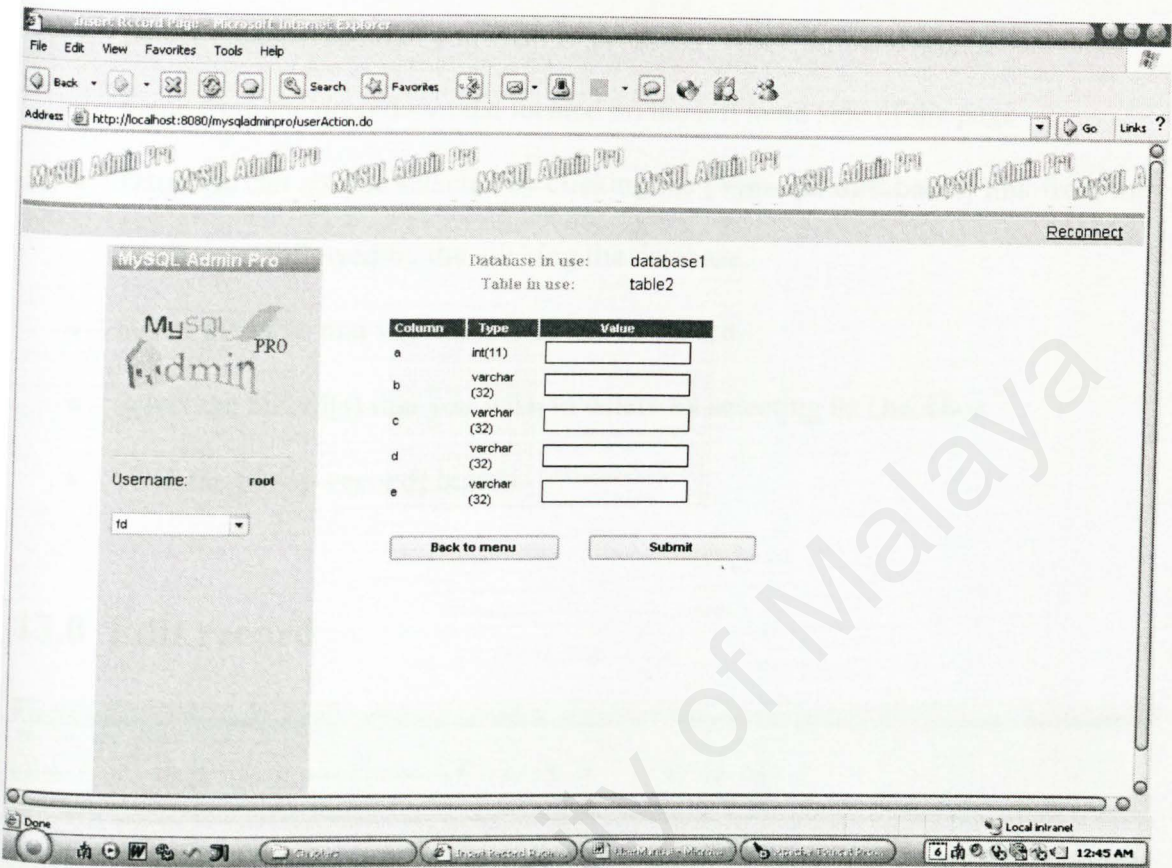
- Enter the table column's name in each **[Field]** of the table
- Select the data type from the **[Type]** drop down list for each column
- Select the length of the data type if necessary by entering the length in the **[Length]** text box. **(Not all data type needs the data length! Error message will be displayed if data type does not match with data length!)**
- Select the character set of the data type if necessary by selecting the **[Charset]** drop down list. **(Not all data type needs character set! Error message will be displayed if data type does not match with character set!)**
- Select **at least** one primary key for the table by selecting the checkbox
- Verify whether the column can have a null value by select the **[Null/Not Null]** drop down list
- Click **[Submit]** button to submit the table

10.0 Drop table



- Select the database that you wish to drop its table, by selecting the database name from the drop down list located on the left hand side of the page.
- Database can also be selected by clicking the **[View all databases]** link from the menu page followed by the clicking the database
- Select the table(s) that you wish to drop by selecting its checkbox.
- Click **[Drop table]** to drop the selected table(s)

11.0 Insert new record

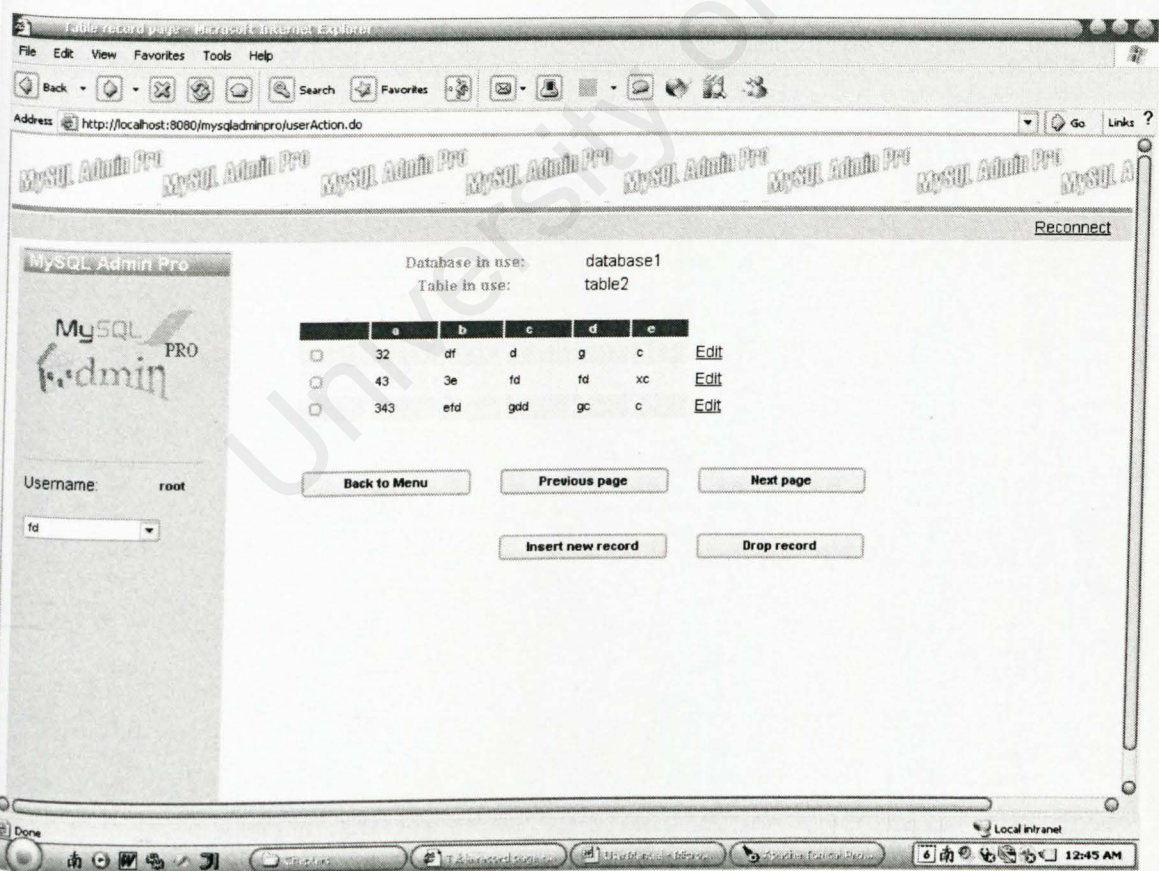


- Select the database that you wish to insert record in, by selecting the database name from the drop down list located on the left hand side of the page.
- Database can also be selected by clicking the **[View all databases]** link from the menu page followed by the clicking the database
- Select the table that you wish to insert your record in.
- Click the **[Insert new record]** button
- Enter the value of the record in the textbox of each column
- Click **[Submit]**

12.0 Delete record

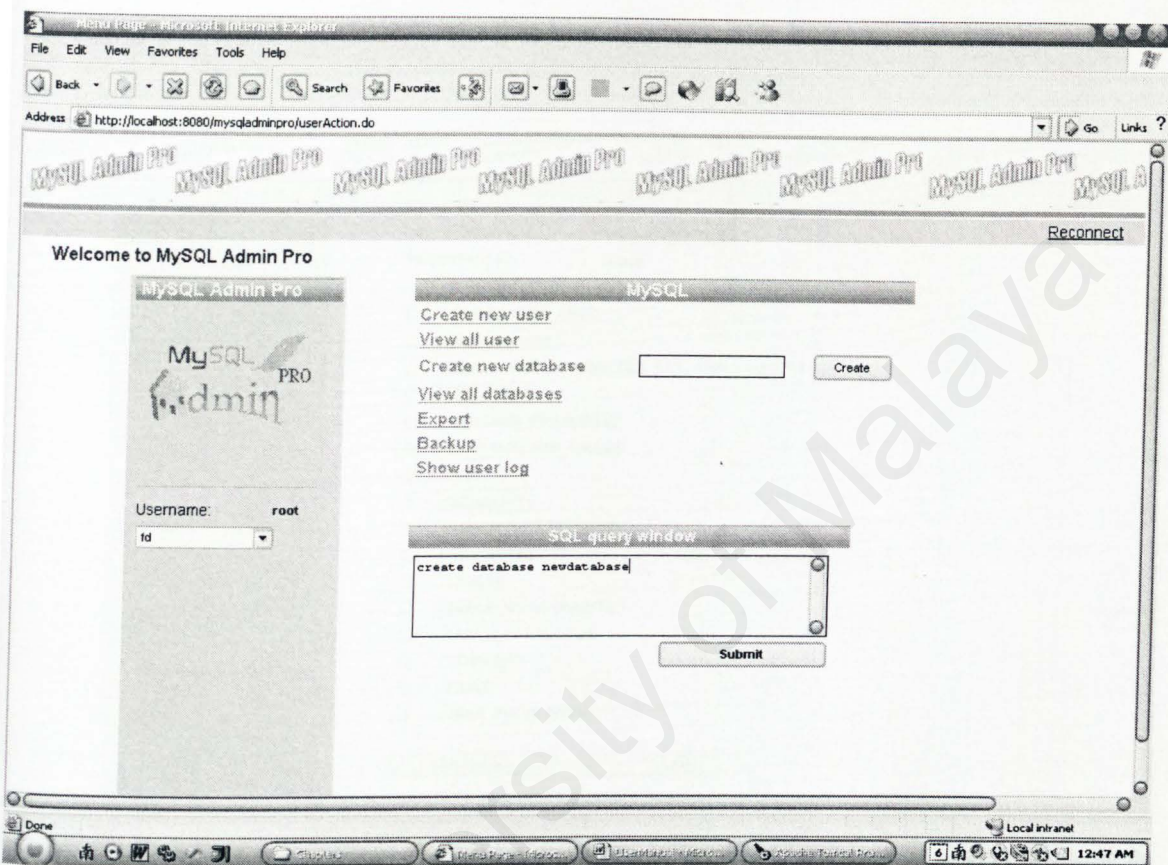
- Select the database that you wish to delete its record, by selecting the database name from the drop down list located on the left hand side of the page.
- Database can also be selected by clicking the **[View all databases]** link from the menu page followed by the clicking the database
- Select the table that you wish to delete its record.
- Select the record(s) that you wish to delete by selecting its checkbox
- Click the **[Drop record]** button

13.0 Edit record



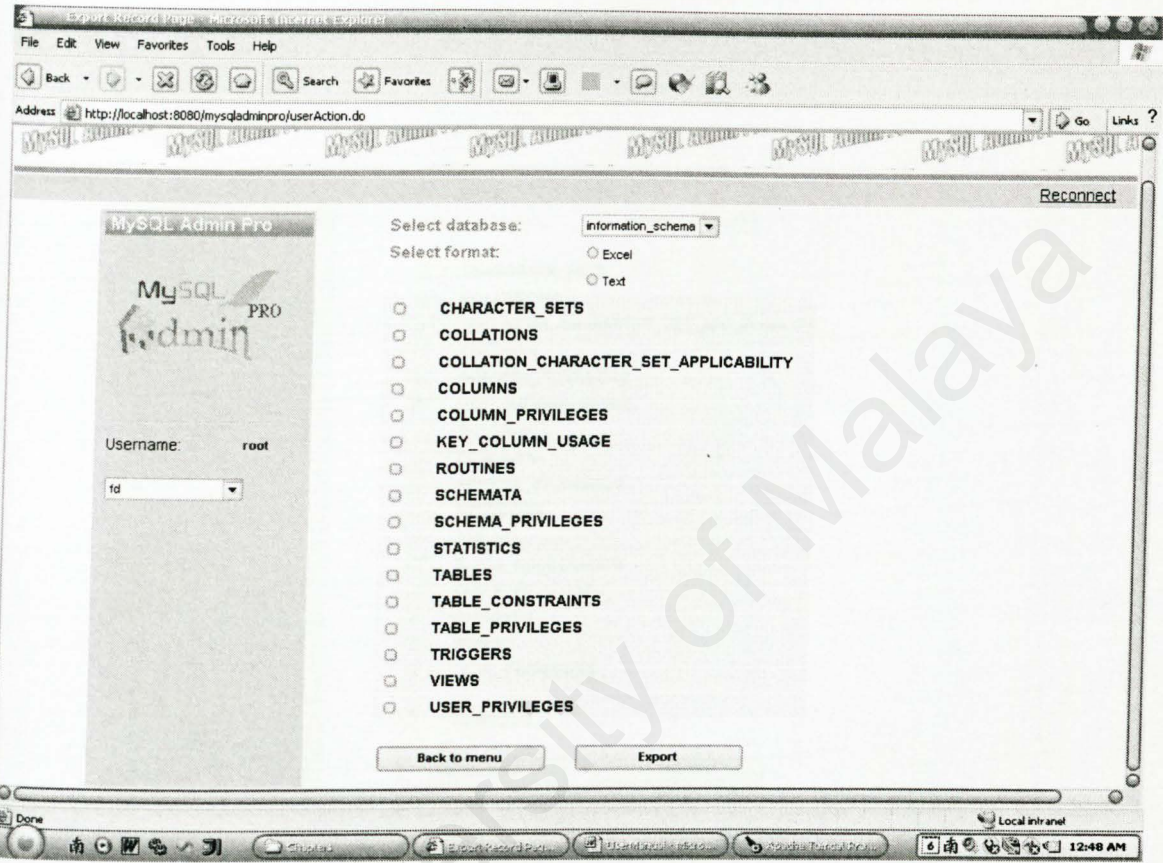
- Select the database that you wish to edit its record, by selecting the database name from the drop down list located on the left hand side of the page.
- Database can also be selected by clicking the [**View all databases**] link from the menu page followed by the clicking the database
- Select the table that you wish to edit its record.
- Click the [**Edit**] link located beside the record that you wish to edit
- Replace the current value of the record with the new value by entering the value in the text box
- Click the [**Submit**] button

14.0 Using the SQL query window



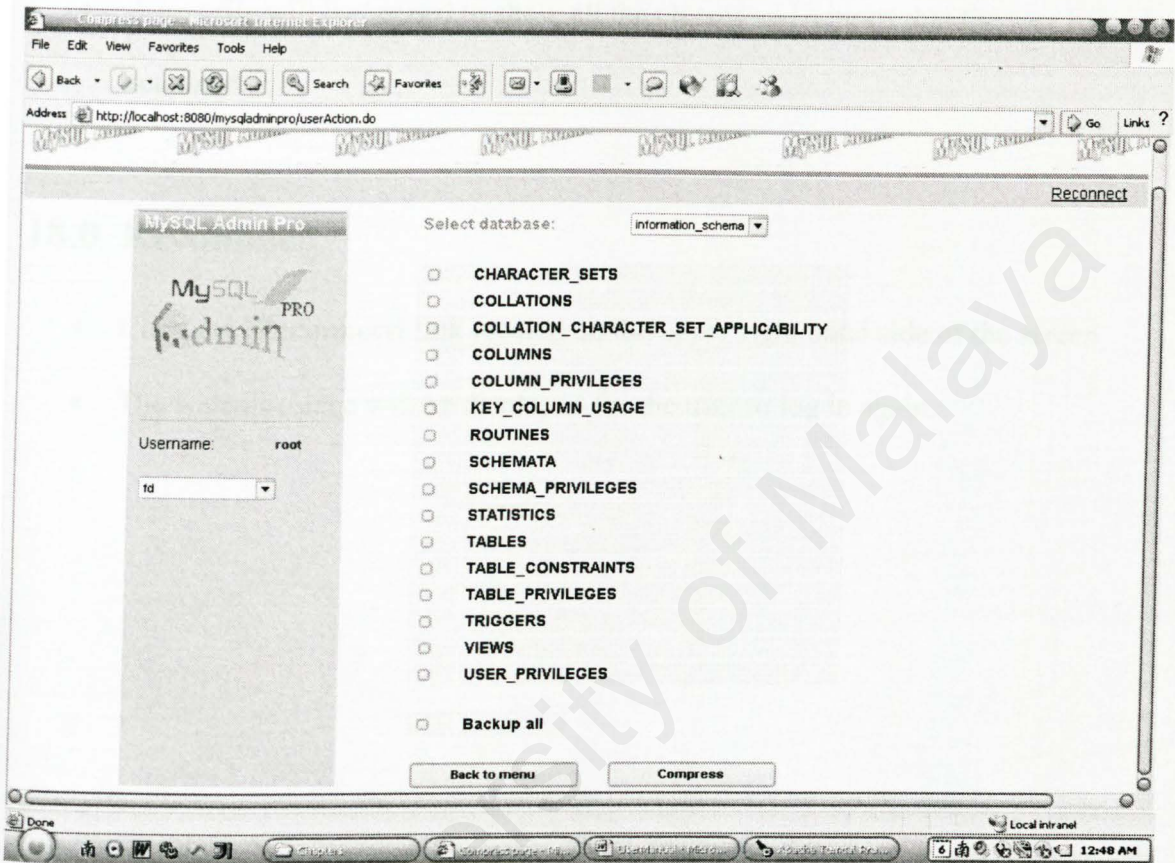
- Enter the SQL command that you need to submit in the query window box
(Query window only supports one query at a time!)
- Click the [Submit] button

15.0 Export table to other file format



- Click the **[Export]** link from the menu page
- Select the database that you wish to export from the drop down list
- Select the format that you want to export (Microsoft Excel or text file)
- Select the table(s) that you wish to export
- Click the **[Export]** button to export the selected table to C:

16.0 Backup and compress database



- Click the **[Backup]** link from the menu page
- Select the database that you wish to backup from the drop down list
- Choose the table(s) that you wish to backup by selecting its checkbox
- Choose **[Backup all]** if you wish to back up the entire database
- Click **[Compress]** to backup the selected database and table(s) to C:

17.0 Show user log/history

- Click the **[Show user log]** link from the menu page
- A text file will be opened to show all the classes and actions that the user has activated

18.0 Reconnect

- Click the **[Reconnect]** link located on the upper right hand side of the screen
- The welcome page will be displayed for the user to log in again.